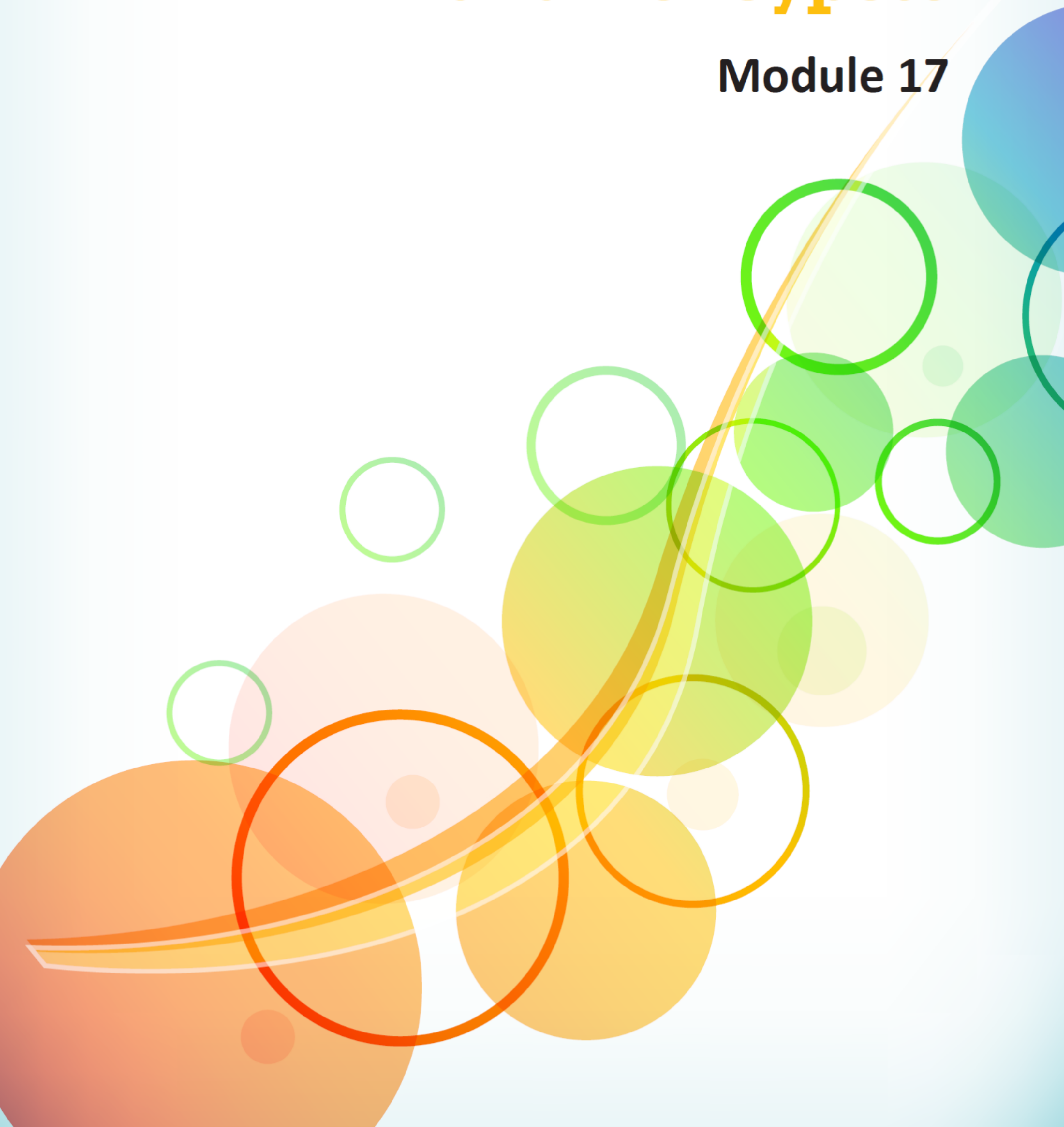
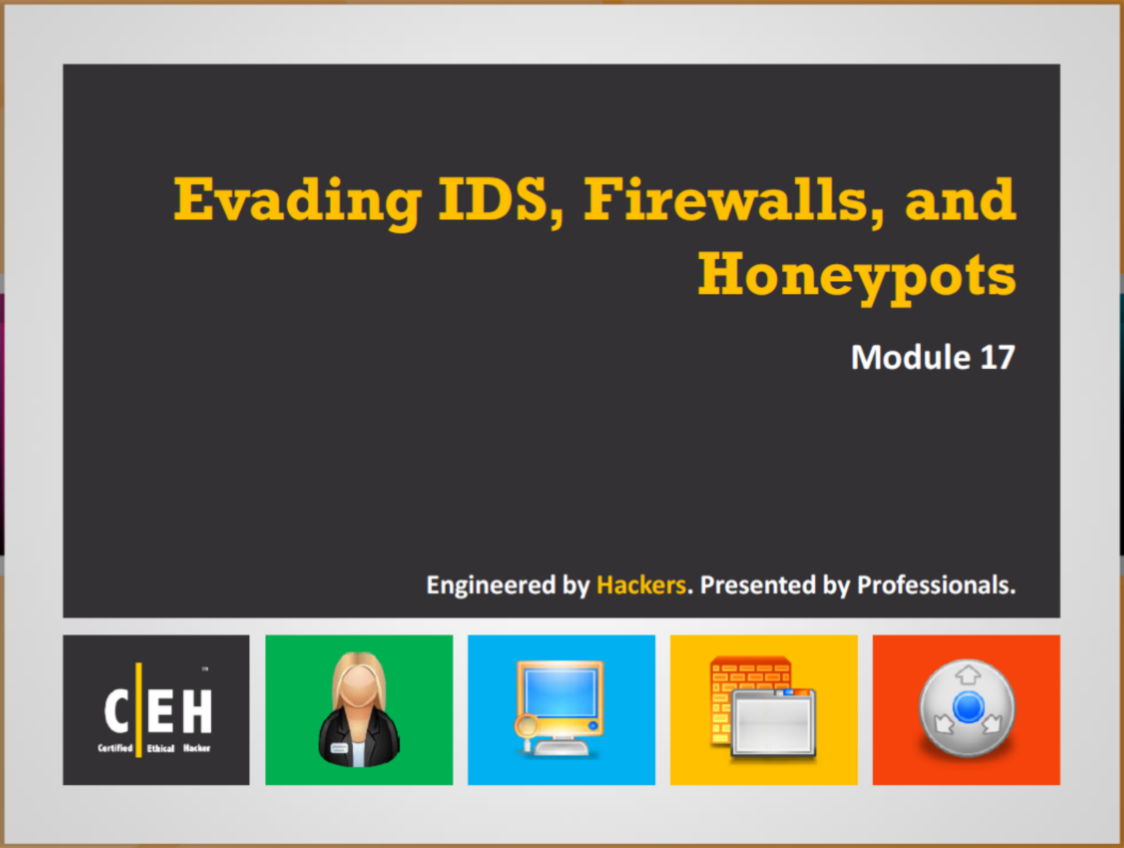
Evading IDS, Firewalls, and Honeypots

Module 17





Evading IDS, Firewalls, and Honeypots

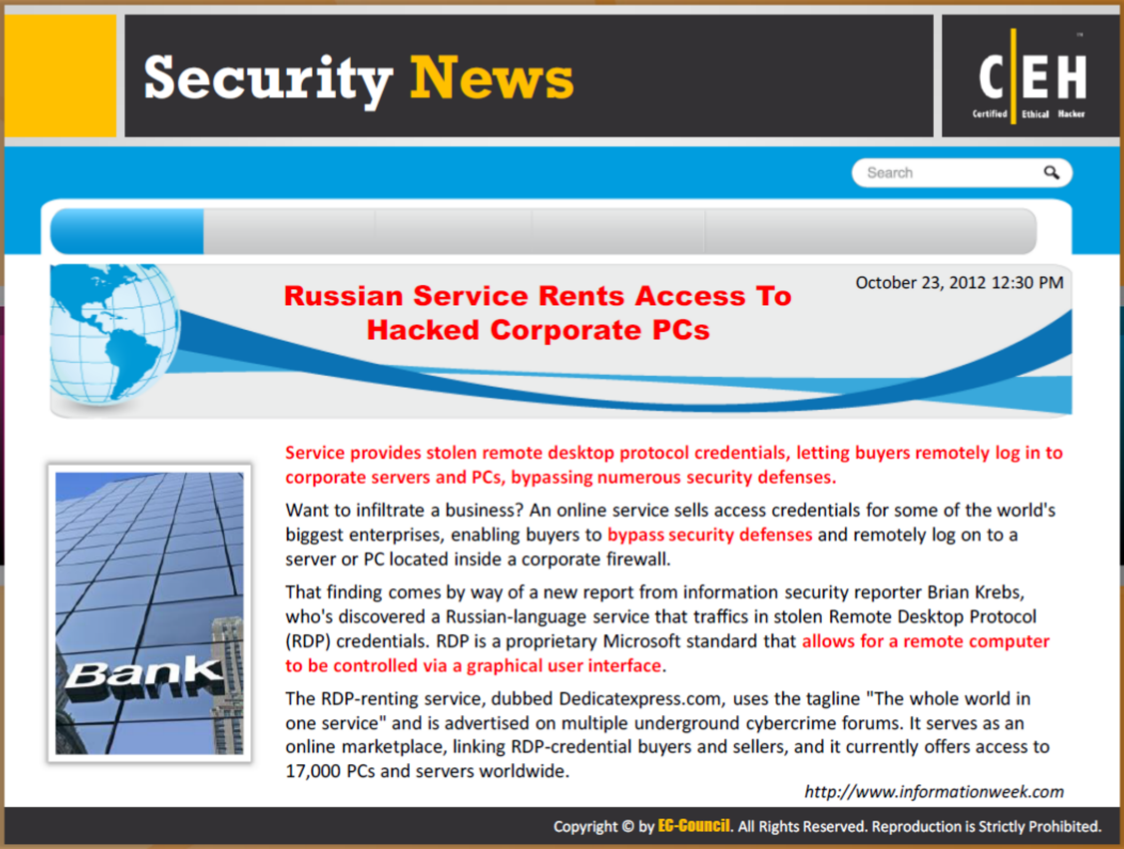
Module 17

Engineered by Hackers. Presented by Professionals.

Ethical Hacking and Countermeasures v8

Module 17: Evading IDS, Firewalls, and Honeypots

Exam 312-50



October 23, 2012 12:30 PM

Russian Service Rents Access To Hacked Corporate PCs

Service provides stolen remote desktop protocol credentials, letting buyers remotely log in to corporate servers and PCs, bypassing numerous security defenses.

Want to infiltrate a business? An online service sells access credentials for some of the world's biggest enterprises, enabling buyers to bypass security defenses and remotely log on to a server or PC located inside a corporate firewall.

That finding comes by way of a new report from information security reporter Brian Krebs, who's discovered a Russian-language service that traffics in stolen Remote Desktop Protocol (RDP) credentials. RDP is a proprietary Microsoft standard that allows for a remote computer to be controlled via a graphical user interface.

The RDP-renting service, dubbed Dedicatexpress.com, uses the tagline "The whole world in one service" and is advertised on multiple underground cybercrime forums. It serves as an online marketplace, linking RDP-credential buyers and sellers, and it currently offers access to 17,000 PCs and servers worldwide.

http://www.informationweek.com

Security News

Russian Service Rents Access To Hacked Corporate PCs

Source: http://www.informationweek.com

Service provides stolen **remote desktop protocol** credentials, letting buyers remotely log in to corporate servers and PCs, bypassing numerous security defenses.

Want to infiltrate a business? An online service sells access credentials for some of the world's biggest enterprises, enabling buyers to bypass security defenses and remotely log on to a server or PC located inside a corporate firewall.

That finding comes by way of a new report from information security reporter Brian Krebs, who's discovered a **Russian-language** service that traffics in stolen Remote Desktop Protocol (RDP) credentials. RDP is a proprietary Microsoft standard that allows for a remote computer to be controlled via a **graphical user interface**.

The RDP-renting service, dubbed Dedicatexpress.com, uses the tagline "The whole world in one service" and is advertised on multiple underground cybercrime forums. It serves as an online marketplace, linking RDP-credential buyers and sellers, and it currently offers access to 17,000 PCs and servers worldwide.

Here's how **Dedicatexpress.com** works: Hackers submit their stolen RDP credentials to the service, which pays them a commission for every rental. According to a screen grab published by Krebs, the top submitters are "lopster," with **12,254 rentals**, followed by "_sz_", with **6,645 rentals**. Interestingly, submitters can restrict what the machines may be used for--for example, specifying that machines aren't to be used to run online gambling operations or PayPal scams, or that they can't be run with **administrator-level** credentials.

New users pay $20 to join the site, after which they can search for available PC and server RDP credentials. Rental prices begin at just a few dollars and vary based on the machine's processor speed, upload and download bandwidth, and the length of time that the machine has been consistently available online.

According to Krebs, the site's managers have said they won't traffic in Russian RDP credentials, suggesting that the site's owners are based in Russia and don't wish to antagonize Russian authorities. According to security experts, Russian law enforcement agencies typically turn a blind eye to cybercrime gangs operating inside their borders, providing they don't target Russians, and that these gangs in fact occasionally assist authorities.

When reviewing the Dedicatexpress.com service, Krebs said he quickly discovered that access was being rented, for $4.55, to a system that was listed in the Internet address space assigned to Cisco, and that several machines in the IP address range assigned to Microsoft's managed hosting network were also available for rent. In the case of Cisco, the RDP credentials--username and password--were both "Cisco." Krebs reported that a Cisco source told him the machine in question was a "bad lab machine."

As the Cisco case highlights, poor username and password combinations, combined with **remote-control applications**, give attackers easy access to corporate networks.

Still, even complex usernames and passwords may not stop attackers. Since Dedicatexpress.com was founded in 2010, it's offered access to about 300,000 different systems in total, according to Krebs. Interestingly, 2010 was the same year that security researchers first discovered the Georbot Trojan application, which scans PCs for signs that remote-control software has been installed and then captures and transmits related credentials to attackers. Earlier this year, security researchers at ESET found that when a **Georbot-infected PC** was unable to contact its designated command-and-control server to receive instructions or transmit stolen data, it instead contacted a server based in the country of Georgia.

When it comes to built-in remote access to Windows machines, RDP technology was first included in the **Windows XP Professional**--but not Home--version of the operating system, and it has been included in every edition of Windows released since then. The current software is dubbed Remote Desktop Services (for servers) and Remote Desktop Connection (for clients).

Might **Windows 8 security improvements** help prevent unauthorized people from logging onto PCs using stolen remote desktop protocol credentials? That's not likely, since Microsoft's new operating system--set to debut later this week--includes the latest version, Remote Desktop Protocol 8.0, built in.

Microsoft has also released a free Windows 8 Remote Desktop application, filed in the "productivity" section of Windows Store. According to Microsoft, "the new Metro-style Remote Desktop app enables you to conveniently access your PC and all of your corporate resources from anywhere."

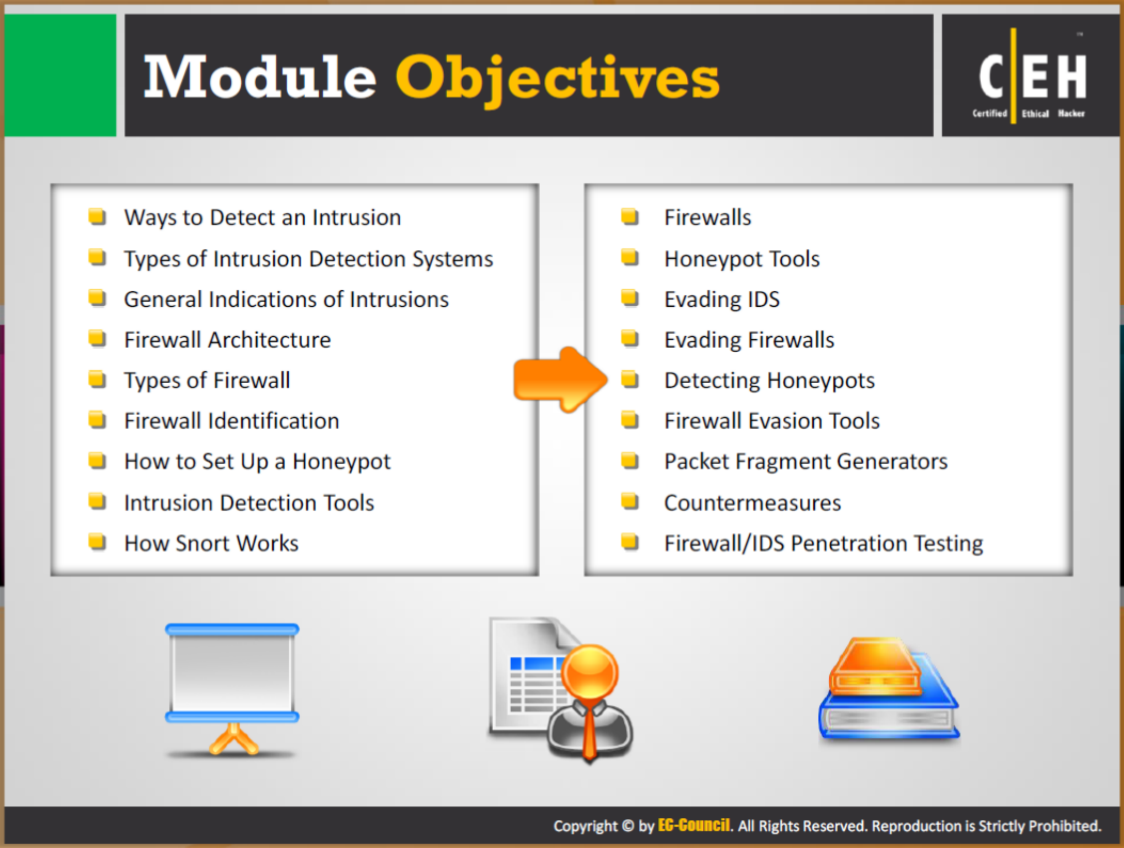
"As many of you already know, a salient feature of Windows Server 2012 and Windows 8 is the ability to deliver a rich user experience for remote desktop users on corporate LAN and WAN networks," read a recent blog post from Shanmugam Kulandaivel, a senior program manager in Microsoft's Remote Desktop Virtualization team.

Despite such capabilities now being built into numerous operating systems--including **Linux** and **Mac OS X**--many security experts recommend deactivating or removing such tools when they're not needed. "Personally, I am a big fan of uninstalling unnecessary software, and it is always sound advice to minimize one's software footprint and related attack surface," said Wolfgang Kandek, CTO of Qualys. He made those comments earlier this year, after the source code for Symantec's pcAnywhere Windows remote-access software was leaked to the Internet by hacktivists. Security experts were concerned that attackers might discover an exploitable zero-day vulnerability in the remote-access code, which would allow them to remotely access any machine that had the software installed.

Copyright © 2012 UBM Tech

By Mathew J.Schwartz

http://www.informationweek.com/security/attacks/russian-service-rents-access-to-hacked-c/240009580



Module Objectives

- Ways to Detect an Intrusion
- Types of Intrusion Detection Systems
- General Indications of Intrusions
- Firewall Architecture
- Types of Firewall
- Firewall Identification
- How to Set Up a Honeypot
- Intrusion Detection Tools
- How Snort Works

- Firewalls
- Honeypot Tools
- Evading IDS
- Evading Firewalls
- Detecting Honeypots
- Firewall Evasion Tools
- Packet Fragment Generators
- Countermeasures
- Firewall/IDS Penetration Testing

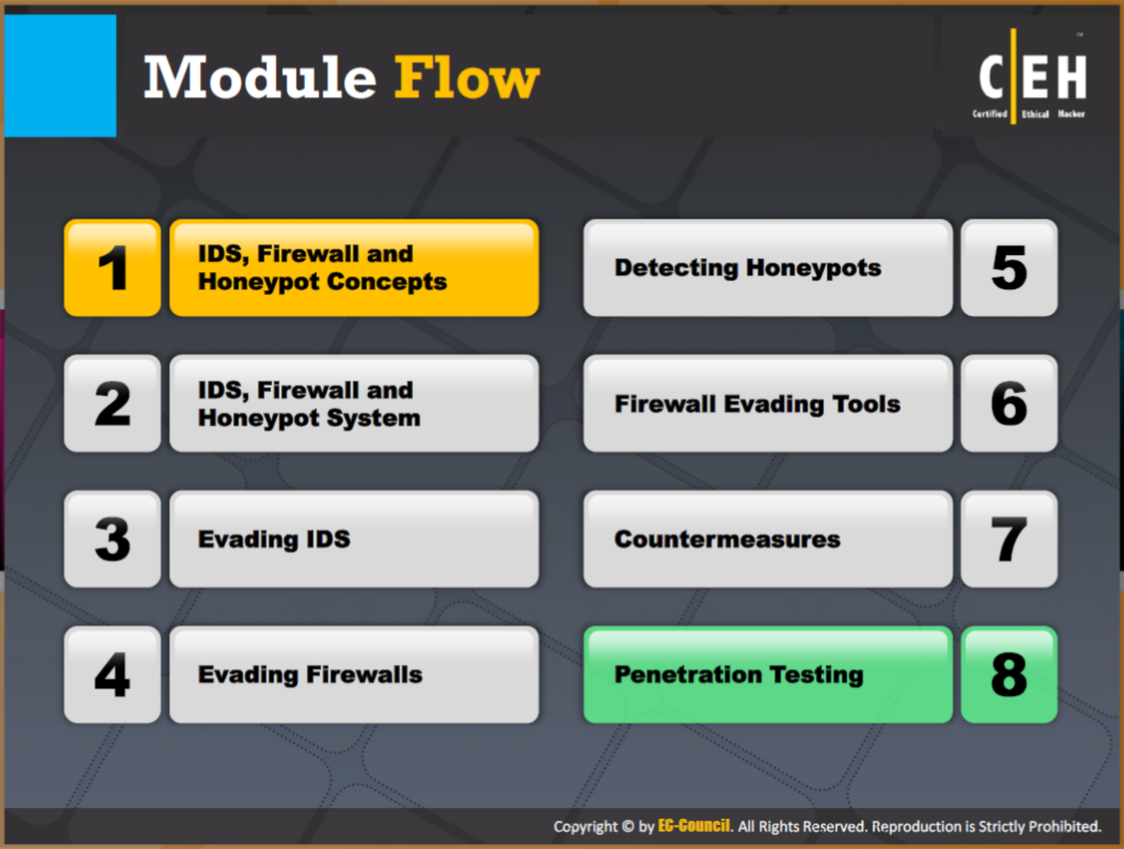
Module Objectives

Today, hacking and computer system attacks are common, making the importance of intrusion detection and active protection all the more relevant. Intrusion detection systems (IDSes), intrusion prevention systems (IPSes), firewalls, and honeypots are the security mechanisms implemented to secure networks or systems. But attackers are able to manage even these security mechanisms and trying to break into the legitimate system or network with the help of various evasion techniques.

This module will familiarize you with:

- Ways to Detect an Intrusion
- Types of Intrusion Detection Systems
- General Indications of Intrusions
- Firewall Architecture
- Types of Firewalls
- Firewall Identification
- How to Set Up a Honeypot
- Intrusion Detection Tools
- How Snort Works

- Firewalls
- Honeypot Tools
- Evading IDSes
- Evading Firewalls
- Detecting Honeypots
- Firewall Evasion Tools
- Packet Fragment Generators
- Countermeasures
- Firewall/IDS Penetration Testing



Module Flow

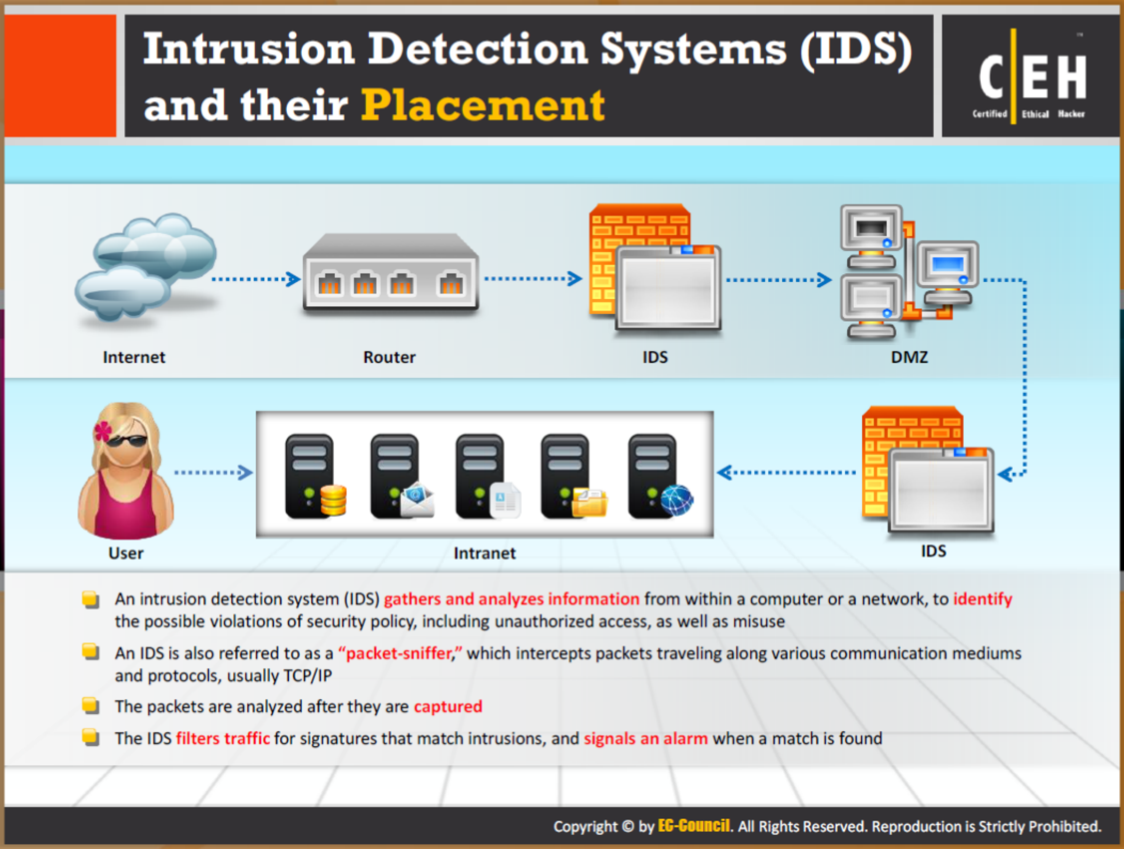
1. IDS, Firewall and Honeypot Concepts
2. IDS, Firewall and Honeypot System
3. Evading IDS
4. Evading Firewalls
5. Detecting Honeypots
6. Firewall Evading Tools
7. Countermeasures
8. Penetration Testing

Module Flow

To understand IDSes, firewalls, and honeypots, evasion techniques used by the attackers to break into the target network or system, it is necessary to understand these mechanisms and how they prevent intrusions and offer protection. So, let us begin with basic IDS, firewall, and honeypot concepts.

IDS, Firewall and Honeypot Concepts	Detecting Honeypots
IDS, Firewall and Honeypot System	Firewall Evading Tools
Evading IDS	Countermeasure
Evading Firewall	Penetration Testing

This section introduces you with the basic IDS, firewall, and honeypot concepts.



Intrusion Detection Systems (IDS) and their Placement

- An intrusion detection system (IDS) gathers and analyzes information from within a computer or a network, to identify the possible violations of security policy, including unauthorized access, as well as misuse
- An IDS is also referred to as a "packet-sniffer," which intercepts packets traveling along various communication mediums and protocols, usually TCP/IP
- The packets are analyzed after they are captured
- The IDS filters traffic for signatures that match intrusions, and signals an alarm when a match is found

Intrusion Detection Systems (IDSes) and their Placement

An intrusion detection system is used to monitor and protect networks or systems for malicious activities. To alert security personnel about intrusions, intrusion detection systems are highly useful. IDSes are used to monitor network traffic. An IDS checks for suspicious activities. It notifies the administrator about intrusions immediately.

- An intrusion detection system (IDS) gathers and analyzes information from within a computer or a network, to identify the possible violations of security policy, including unauthorized access, as well as misuse
- An IDS is also referred to as a "packet-sniffer," which intercepts packets traveling along various communication mediums and protocols, usually TCP/IP
- The packets are analyzed after they are captured
- An IDS evaluates a suspected intrusion once it has taken place and signals an alarm

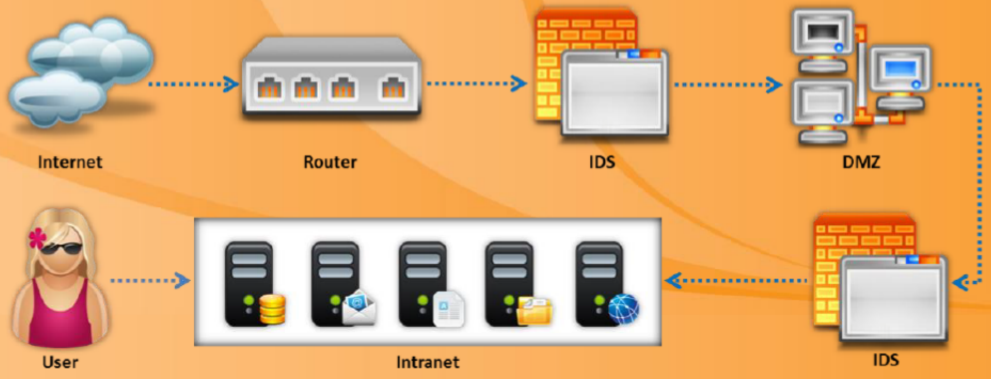
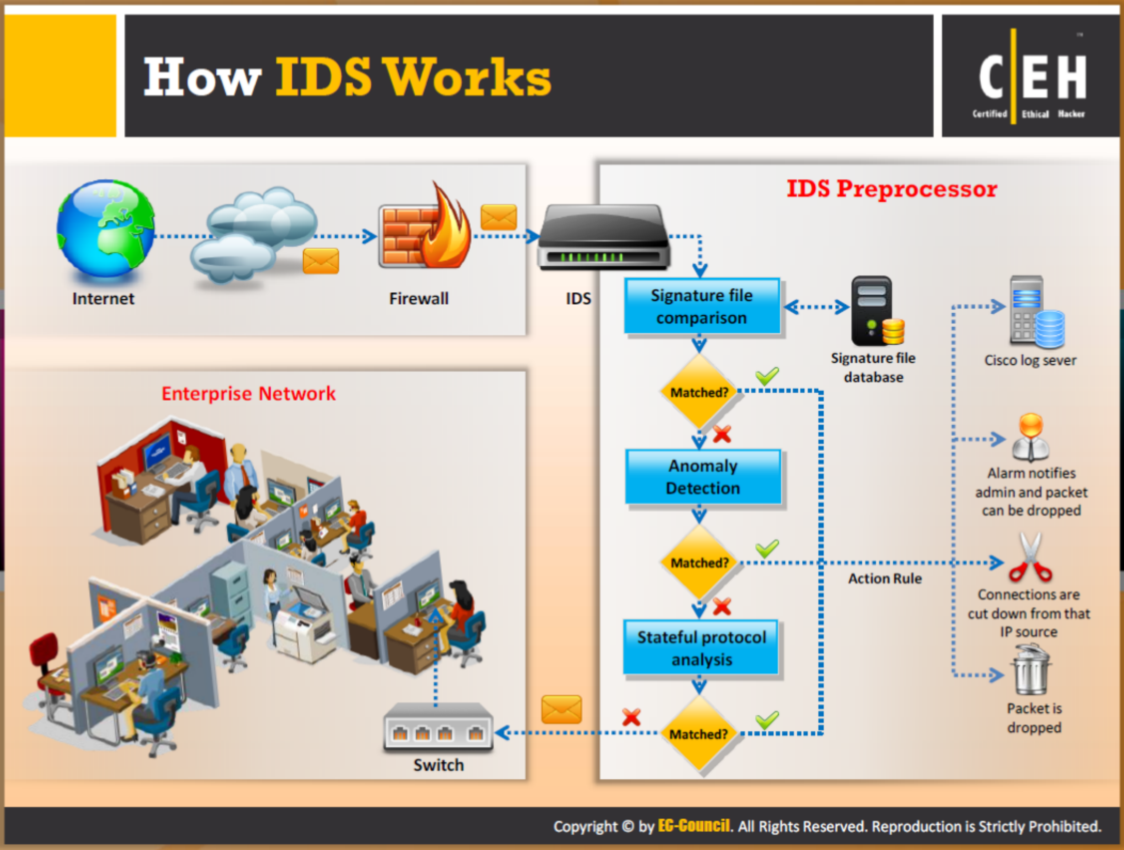

Internet
Router
IDS
DMZ
User
Intranet
IDS

FIGURE 17.1: Intrusion Detection Systems (IDSes) and their Placement



How IDS Works

C|EH
Certified Ethical Hacker
Internet
Firewall
IDS
Enterprise Network
Switch
IDS Preprocessor
Signature file comparison
Signature file database
Matched?
Anomaly Detection
Matched?
Stateful protocol analysis
Matched?
Action Rule
Cisco log sever
Alarm notifies admin and packet can be dropped
Connections are cut down from that IP source
Packet is dropped

How an IDS Works

The main purposes of IDSes are that they not only **prevent intrusions** but also alert the **administrator immediately** when the attack is still going on. The administrator could identify methods and techniques being used by the intruder and also the source of attack.

An IDS works in the following way:

- IDSes have sensors to **detect signatures** and some advanced IDSes have behavioral activity detection to determine malicious behavior. Even if signatures don't match this activity detection system can alert administrators about possible attacks.
- If the signature matches, then it moves to the next step or the **connections are cut down** from that IP source, the packet is dropped, and the alarm notifies the admin and the packet can be dropped.
- Once the signature is matched, then sensors pass on **anomaly detection**, whether the received packet or request matches or not.
- If the packet passes the anomaly stage, then stateful protocol analysis is done. After that through switch the packets are passed on to the network. If anything mismatches again, the connections are cut down from that **IP source**, the packet is dropped, and the alarm notifies the admin and packet can be dropped.

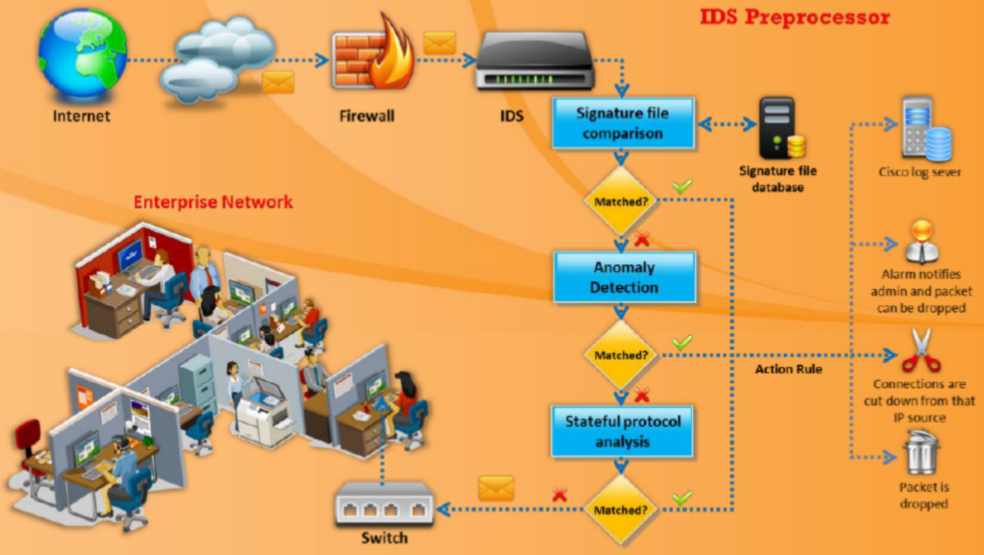
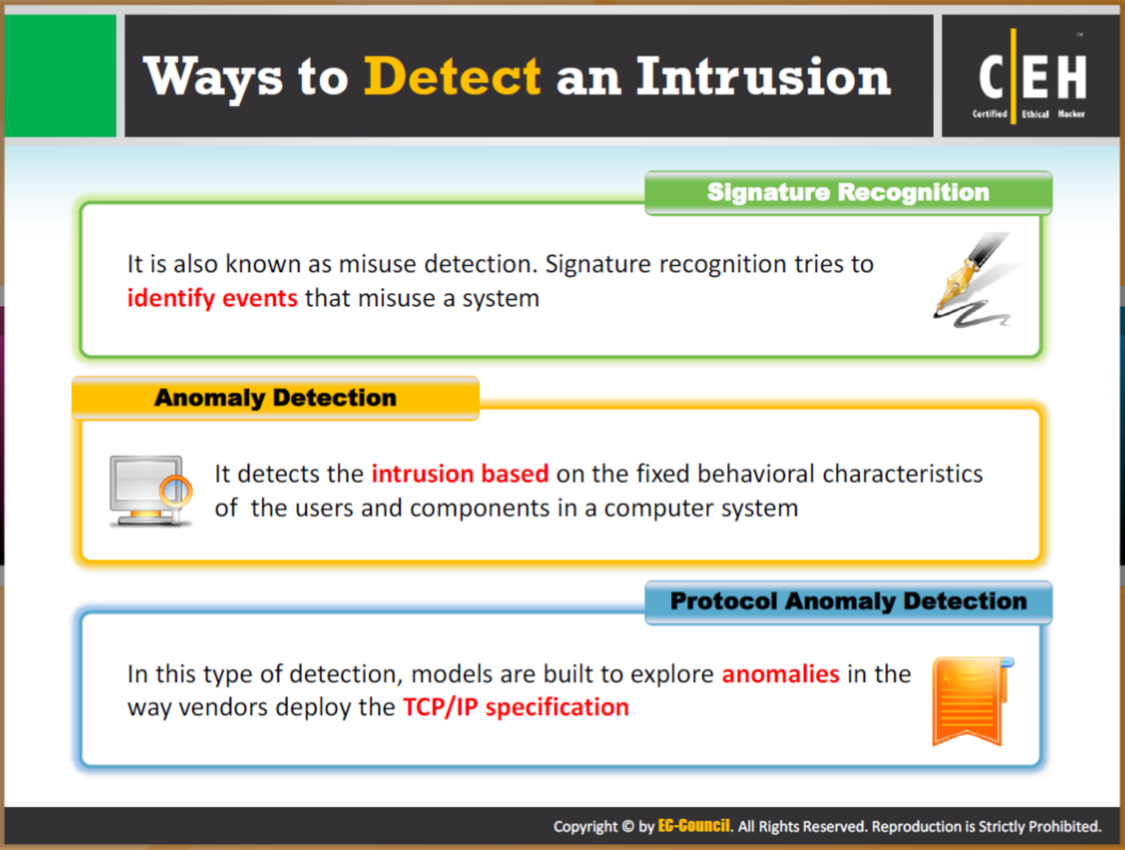

IDS Preprocessor
Internet
Firewall
IDS
Signature file comparison
Signature file database
Cisco log sever
Matched?
Enterprise Network
Anomaly Detection
Alarm notifies admin and packet can be dropped
Matched?
Action Rule
Connections are cut down from that IP source
Stateful protocol analysis
Packet is dropped
Matched?
Switch

FIGURE 17.2: How an IDS Works



Ways to **Detect** an Intrusion

Signature Recognition

It is also known as misuse detection. Signature recognition tries to **identify events** that misuse a system

Anomaly Detection

It detects the **intrusion based** on the fixed behavioral characteristics of the users and components in a computer system

Protocol Anomaly Detection

In this type of detection, models are built to explore **anomalies** in the way vendors deploy the **TCP/IP specification**

Ways to Detect an Intrusion

An intrusion is detected in three ways.

Signature Detection

Signature recognition is also known as **misuse detection**. It tries to identify events that indicate an abuse of a system. It is achieved by creating models of intrusions. Incoming events are compared with intrusion models to make a detection decision. While creating signatures, the model must detect an attack without disturbing the normal traffic on the system. Attacks, and only attacks, should match the model or else false alarms can be generated.

- The simplest form of signature recognition uses simple pattern matching to compare the network packets against binary signatures of known attacks. A binary signature may be defined for a specific portion of the packet, such as the **TCP flags**.
- Signature recognition can detect known **attacks**. However, there is a possibility that other packets that match might represent the signature, triggering bogus signals. Signatures can be customized so that even well-informed users can create them.
- Signatures that are formed improperly may trigger **bogus signals**. In order to detect misuse, the number of signatures required is huge. The more the signatures, the more

attacks can be detected, though traffic may incorrectly match with the signatures, reducing the performance of the system.

- The bandwidth of the network is consumed with the increase in the signature database. As the signatures are compared against those in the database, there is a probability that the maximum number of comparisons cannot be made, resulting in certain packets being dropped.
- New virus attacks such as **ADMutate** and **Nimda** create the need for multiple signatures for a single attack. Changing a single bit in some attack strings can invalidate a signature and create the need for an entirely new signature.
- Despite problems with signature-based intrusion detection, such systems are popular and work well when configured correctly and monitored closely

Anomaly Detection

Anomaly detection is otherwise called "**not-use detection**." Anomaly detection differs from the signature recognition model. The model consists of a database of anomalies. Any event that is identified with the **database** in considered an anomaly. Any deviation from normal use is **labeled an attack**. Creating a model of normal use is the most difficult task in creating an anomaly detector.

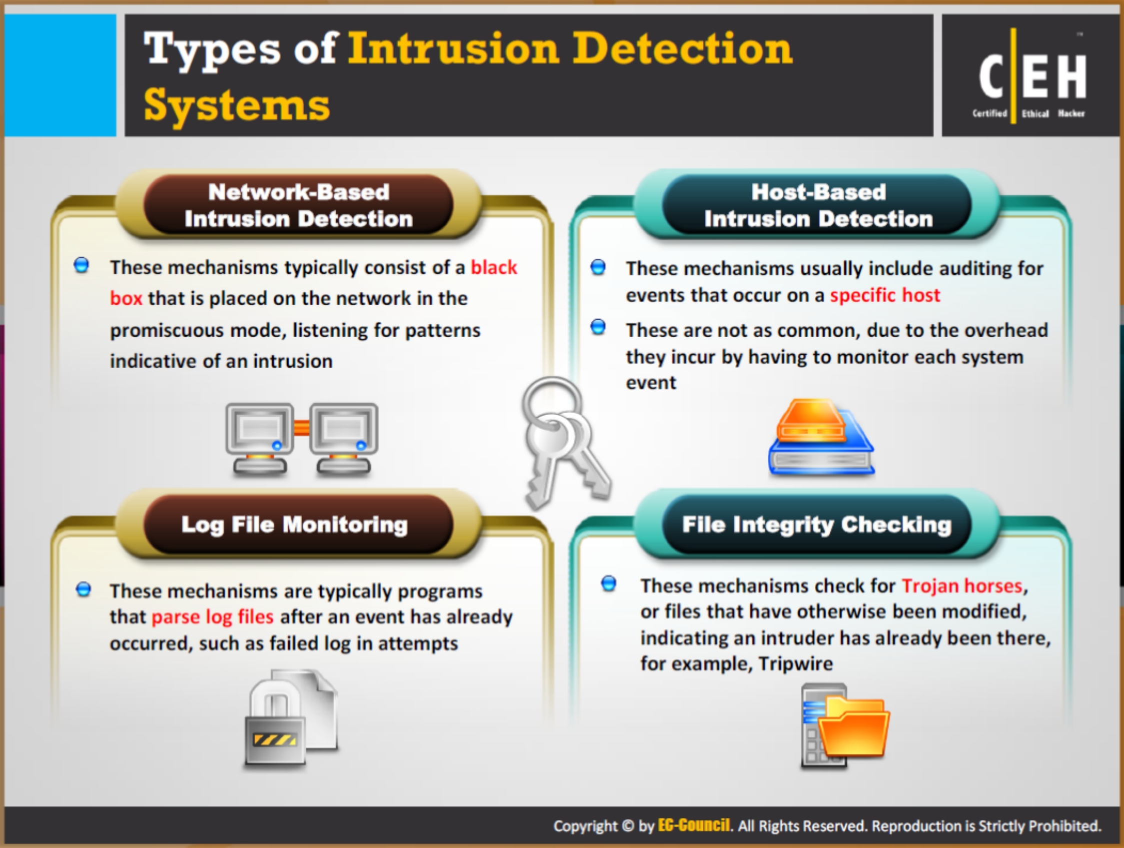
- In the traditional method of anomaly detection, important data is kept for checking variations in network traffic for the model. However, in reality, there is less variation in **network traffic** and too many statistical variations making these models imprecise; some events labeled as anomalies might only be irregularities in network usage.
- In this type of approach, the inability to instruct a model thoroughly on the normal network is of grave concern. These models should be trained on the specific network that is to be policed.

Protocol Anomaly Detection

Protocol anomaly detection is based on the anomalies specific to a protocol. This model is integrated into the **IDS model** recently. It identifies the **TCP/IP protocol** specific flaws in the network. Protocols are created with specifications, known as RFCs, for dictating proper use and communication. The protocol anomaly detector can identify new attacks.

- There are new attack methods and exploits that violate protocol standards being discovered frequently.
- The pace at which the **malicious signature attacker** is growing is incredibly fast. But the network protocol, in comparison, is well defined and changing slowly. Therefore, the signature database must be updated frequently to detect attacks.
- Protocol anomaly detection systems are easier to use because they require no signature updates

- Protocol anomaly detectors are different from the traditional IDS in how they present alarms.
- The best way to present alarms is to explain which part of the state system was compromised. For this, the IDS operators have to have a thorough knowledge of the protocol design; the best way is the documentation provided by the IDS.



Types of Intrusion Detection Systems

Network-Based Intrusion Detection

- These mechanisms typically consist of a black box that is placed on the network in the promiscuous mode, listening for patterns indicative of an intrusion

Host-Based Intrusion Detection

- These mechanisms usually include auditing for events that occur on a specific host
- These are not as common, due to the overhead they incur by having to monitor each system event

Log File Monitoring

- These mechanisms are typically programs that parse log files after an event has already occurred, such as failed log in attempts

File Integrity Checking

- These mechanisms check for Trojan horses, or files that have otherwise been modified, indicating an intruder has already been there, for example, Tripwire

Types of Intrusion Detection Systems

Basically there are four types of intrusion detection systems are available. They are:

Network-based Intrusion Detection

The NIDS checks every packet entering the network for the presence of **anomalies** and **incorrect data**. Unlike the firewalls that are confined to the filtering of data packets with vivid malicious content, the NIDS checks every packet thoroughly. An **NIDS captures** and inspects all traffic, regardless of whether it is permitted. Based on the content, at either the IP or application-level, an alert is generated. Network-based intrusion detection systems tend to be more distributed than **host-based IDSes**. The NIDS is basically designed to identify the anomalies at the router- and host-level. The NIDS audits the information contained in the data packets, logging information of malicious packets. A threat level is assigned to each risk after the data packets are received. The threat level enables the security team to be on alert. These mechanisms typically consist of a **black box** that is placed on the network in the promiscuous mode, listening for patterns indicative of an intrusion.

Host-based Intrusion Detection

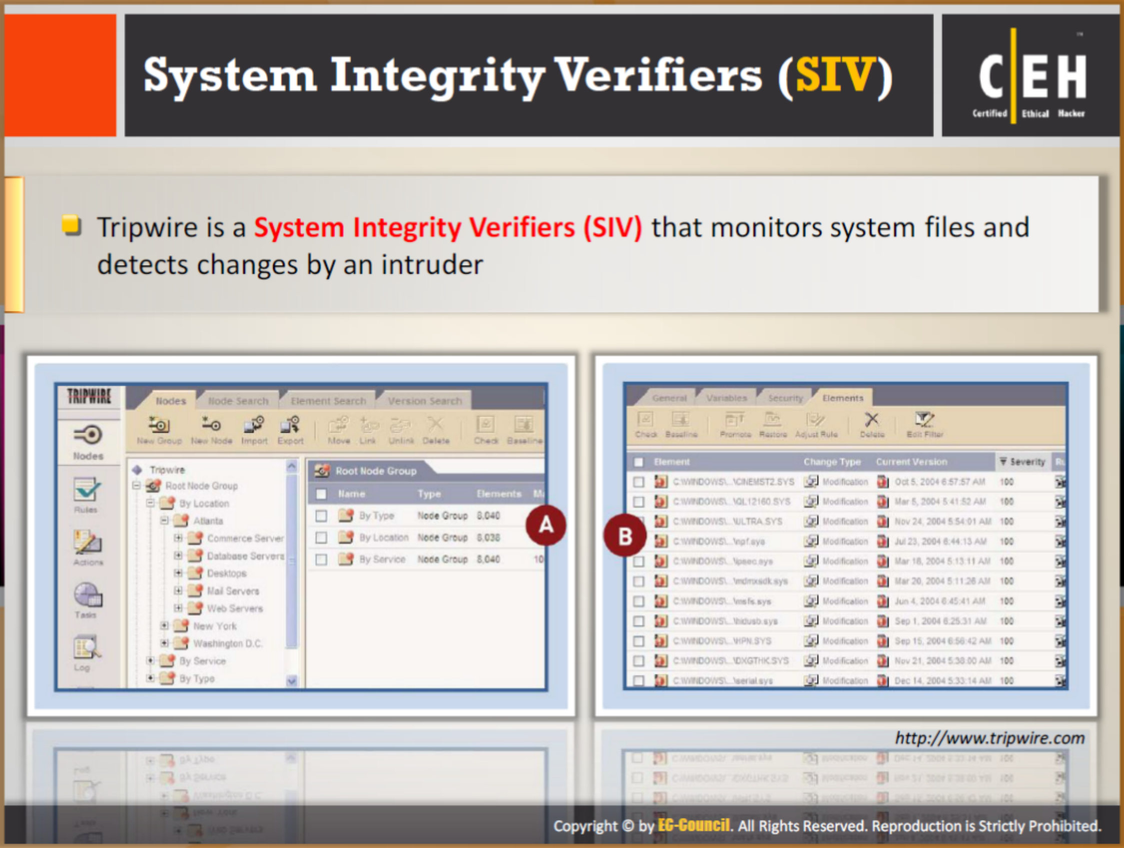
In the host-based system, the **IDS analyzes** each system's behavior. The HIDS can be installed on any system ranging from a desktop PC to a server. The HIDS is more versatile than the NIDS. One example of a host-based system is a program that operates on a system and receives application or operating system audit logs. These programs are highly effective for detecting insider abuses. Residing on the trusted network systems themselves, they are close to the network's authenticated users. If one of these users attempts unauthorized activity, host-based systems usually detect and collect the most pertinent information promptly. In addition to detecting unauthorized insider activity, host-based systems are also effective at detecting unauthorized file modification. HIDSes are more focused on changing aspects of the local systems. HIDS is also more platform-centric, with more focus on the Windows OS, but there are other **HIDSes** for **UNIX platforms**. These mechanisms usually include auditing for events that occur on a specific host. These are not as common, due to the overhead they incur by having to monitor each system event

Log File Monitoring

A Log File Monitor (LFM) monitors log files created by network services. The **LFT IDS** searches through the logs and identifies malicious events. In a similar manner to NIDS, these systems look for patterns in the log files that suggest an intrusion. A typical example would be parsers for **HTTP server** log files that look for intruders who try well-known security holes, such as the "**phf**" attack. An example is swatch. These mechanisms are typically programs that parse log files after an event has already occurred, such as failed log in attempts.

File Integrity Checking

These mechanisms check for Trojan horses, or files that have otherwise been modified, indicating an intruder has already been there, for example, Tripwire.



System Integrity Verifiers (SIV)

- Tripwire is a **System Integrity Verifiers (SIV)** that monitors system files and detects changes by an intruder

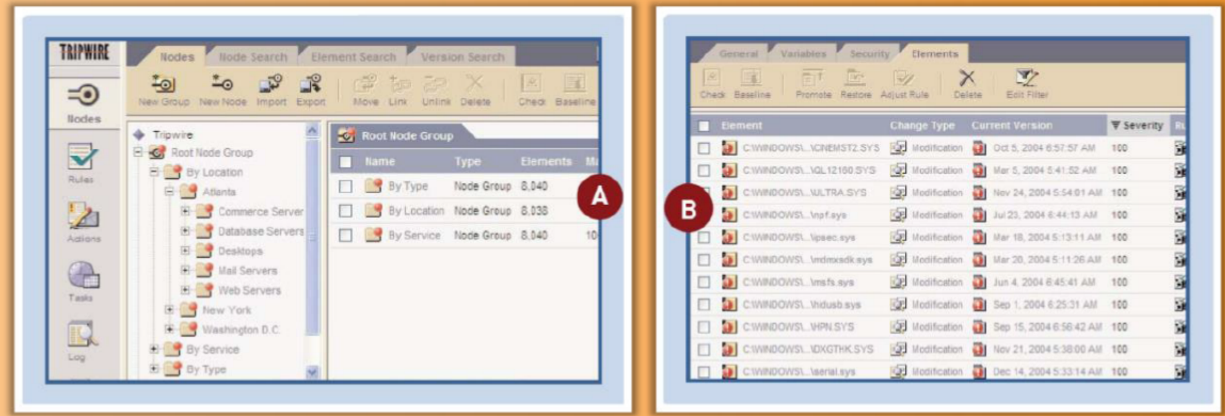
http://www.tripwire.com

C|EH
Certified Ethical Hacker
A
B

System Integrity Verifiers (SIV)

Source: http://www.tripwire.com

A System Integrity Verifier (SIV) **monitors system files** to determine whether an intruder has changed the files. An integrity monitor watches key system objects for changes. For example, a basic integrity monitor uses system files, or registry keys, to track changes by an intruder. Although they have limited functionality, integrity monitors can add an additional layer of protection to other forms of **intrusion detection**.

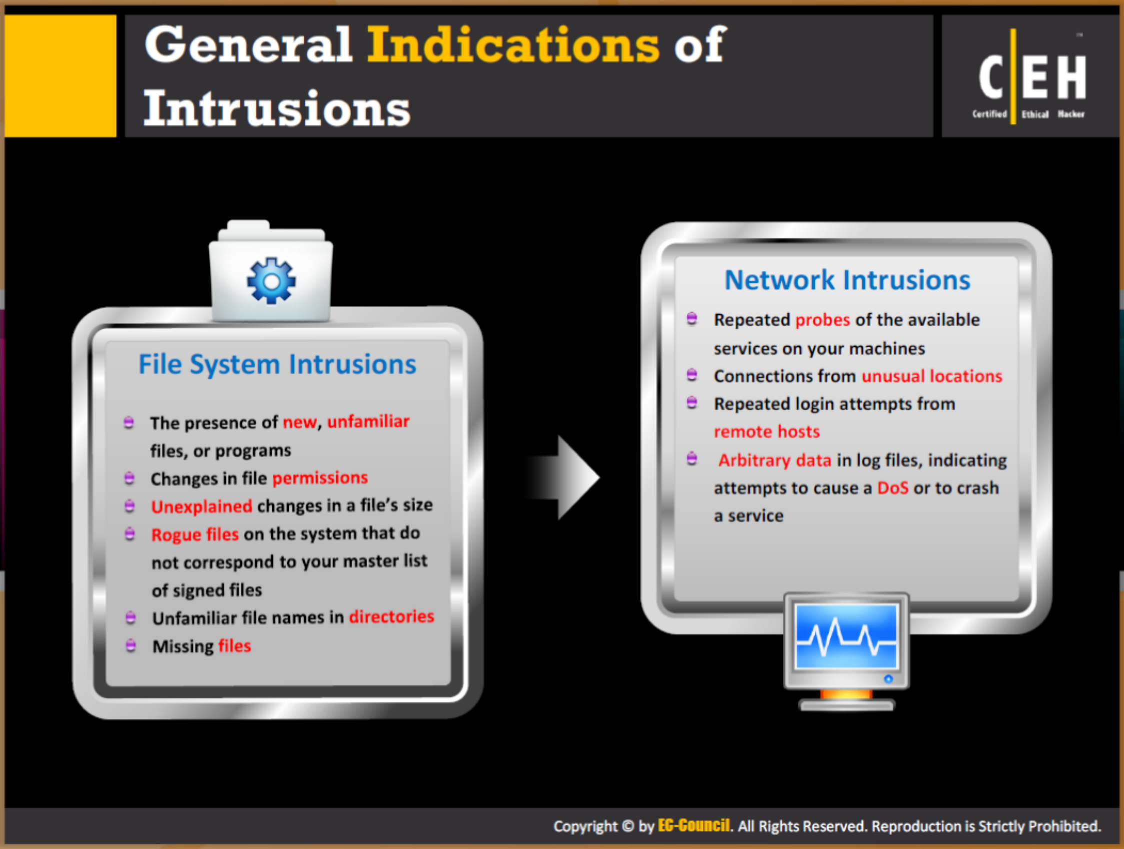


TRIPWIRE
Nodes
Node Search
Element Search
Version Search
New Group
New Node
Import
Export
Move
Link
Unlink
Delete
Check
Baseline
Nodes
Rules
Actions
Tasks
Log
Tripwire
Root Node Group
By Location
Atlanta
Commerce Server
Database Servers
Desktops
Mail Servers
Web Servers
New York
Washington D.C.
By Service
By Type
Root Node Group
A
General
Variables
Security
Elements
Check
Baseline
Promote
Restore
Adjust Rule
Delete
Edit Filter
B

Name	Type	Elements
☐ By Type	Node Group	8,040
☐ By Location	Node Group	8,038
☐ By Service	Node Group	8,040

Element	Change Type	Current Version	Severity
☐ C:\WINDOWS\...\CNEMST2.SYS	Modification	Oct 5, 2004 6:57:57 AM	100
☐ C:\WINDOWS\...\QL12160.SYS	Modification	Mar 5, 2004 5:41:52 AM	100
☐ C:\WINDOWS\...\ULTRA.SYS	Modification	Nov 24, 2004 5:54:01 AM	100
☐ C:\WINDOWS\...\npf.sys	Modification	Jul 23, 2004 6:44:13 AM	100
☐ C:\WINDOWS\...\ipsec.sys	Modification	Mar 18, 2004 5:13:11 AM	100
☐ C:\WINDOWS\...\mdmxsdk.sys	Modification	Mar 20, 2004 5:11:26 AM	100
☐ C:\WINDOWS\...\msfs.sys	Modification	Jun 4, 2004 6:45:41 AM	100
☐ C:\WINDOWS\...\hidusb.sys	Modification	Sep 1, 2004 6:25:31 AM	100
☐ C:\WINDOWS\...\HPN.SYS	Modification	Sep 15, 2004 6:56:42 AM	100
☐ C:\WINDOWS\...\DXGTHK.SYS	Modification	Nov 21, 2004 5:38:00 AM	100
☐ C:\WINDOWS\...\serial.sys	Modification	Dec 14, 2004 5:33:14 AM	100

FIGURE 17.3: System Integrity Verifiers (SIV) Screenshot



General Indications of Intrusions

File System Intrusions

- The presence of new, unfamiliar files, or programs
- Changes in file permissions
- Unexplained changes in a file's size
- Rogue files on the system that do not correspond to your master list of signed files
- Unfamiliar file names in directories
- Missing files

Network Intrusions

- Repeated probes of the available services on your machines
- Connections from unusual locations
- Repeated login attempts from remote hosts
- Arbitrary data in log files, indicating attempts to cause a DoS or to crash a service

General Indications of Intrusions

Following are the general indications of intrusions:

File System Intrusions

By observing the system files, you can identify the presence of an intruder. The system files record the activities of the system. Any modification or deletion in the file attributes or the file itself is a sign that the system was a target of attack:

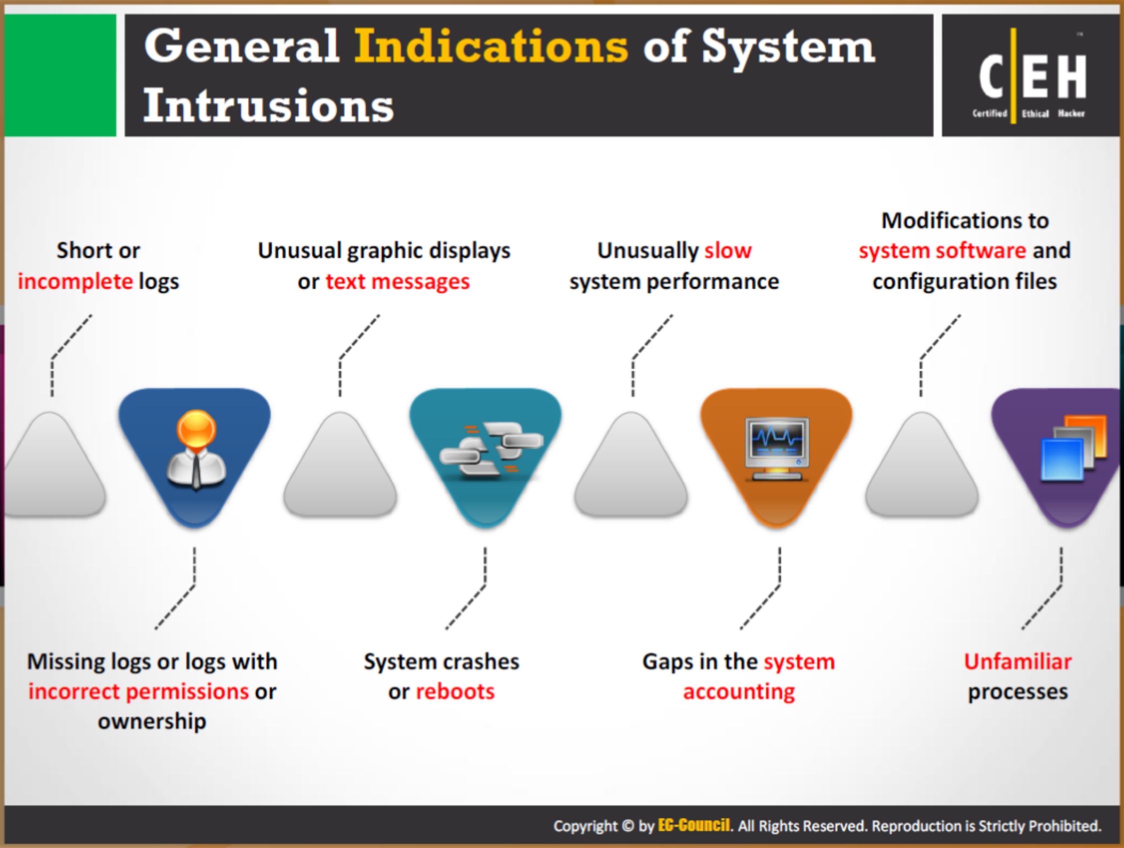
- If you find new, **unknown files/programs** on your system, then there is a possibility that your system has been intruded. The system can be compromised to the point that it can in turn **compromise other systems** in your network.
- When an intruder gains access to a system, he or she tries to escalate privileges to gain administrative access. When the intruder obtains the Administrator privilege, he or she changes the file permissions, for example, from **Read-Only to Write**.
- Unexplained modifications in file size are also an indication of an attack. Make sure you analyze all of your system files.
- Presence of rogue suid and sgid files on your Linux system that do not match your master list of suid and sgid files could indicate an attack.

- You can identify unfamiliar file names in directories, including executable files with strange extensions and double extensions.
- Missing files are also sign of a probable intrusion/attack.

Network Intrusions

General indications of network intrusions:

- Sudden increase in bandwidth consumption is an indication of intrusion.
- Repeated probes of the available services on your machines.
- Connection requests from IPs other than those in the network range are an indication that an **unauthenticated user (intruder)** is attempting to connect to the network.
- You can identify repeated attempts to log in from remote machines.
- Arbitrary log data in log files indicates attempts of **denial-of-service attacks**, bandwidth consumption, and distributed denial-of-service attacks.



General Indications of System Intrusions

Short or incomplete logs

Unusual graphic displays or text messages

Unusually slow system performance

Modifications to system software and configuration files

Missing logs or logs with incorrect permissions or ownership

System crashes or reboots

Gaps in the system accounting

Unfamiliar processes

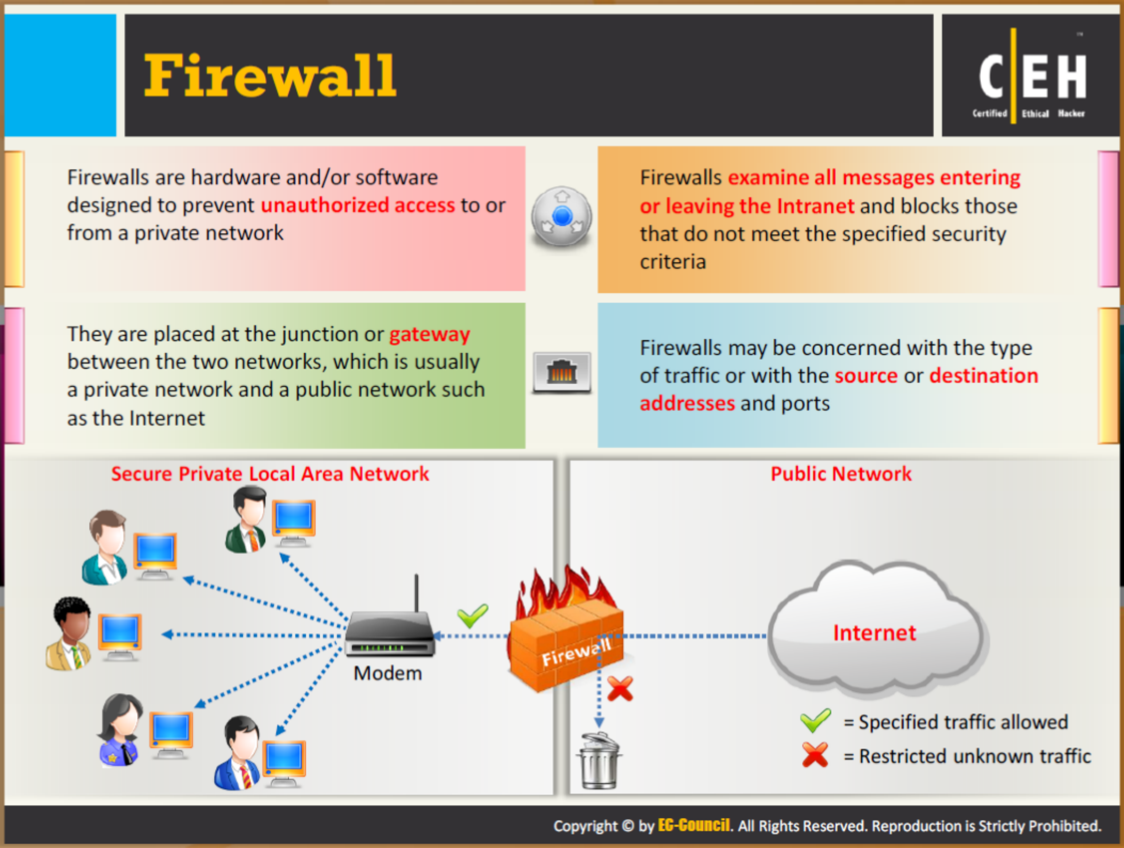
General Indications of System Intrusions

To check whether the system is attacked, you need to check certain parameters that clearly indicate the presence of an intruder on the system. When an intruder attempts to break into the system, he or she attempts to hide his or her presence by modifying certain system files and configurations that indicate intrusion.

Certain signs of intrusion include:

- System's failure in identifying valid user
- Active access to unused logins
- Logins during non-working hours
- New user accounts other than the accounts created
- Modifications to system software and configuration files using Administrator access and the presence of hidden files
- Gaps in system audit files, which indicate that the system was idle for that particular time; he gaps actually indicate that the intruder has attempted to erase the audit tracks
- The system's performance decreases drastically, consuming CPU time
- System crashes suddenly and reboots without user intervention

- The system logs are too short and incomplete
- Timestamps of system logs are modified to include strange inputs
- Permissions on the logs are changed, including the ownership of the logs
- System logs are deleted
- Systems performance is abnormal, the system responds in unfamiliar ways
- Unknown processes are identified on the system
- Unusual display of graphics, pop-ups, and text messages observed on the system



Firewall

Firewalls are hardware and/or software designed to prevent unauthorized access to or from a private network

Firewalls examine all messages entering or leaving the Intranet and blocks those that do not meet the specified security criteria

They are placed at the junction or gateway between the two networks, which is usually a private network and a public network such as the Internet

Firewalls may be concerned with the type of traffic or with the source or destination addresses and ports

C|EH Certified Ethical Hacker
Secure Private Local Area Network
Public Network
Modem
Firewall
Internet
✓ = Specified traffic allowed
✗ = Restricted unknown traffic

Firewalls

A firewall is a set of related programs located at the network gateway server that protects the resources of a private network from users on other networks. Firewalls are a set of tools that monitor the flow of traffic between networks. A firewall, placed at the network level and working closely with a router, filters all network packets to determine whether or not to forward them toward their destinations. A firewall is often installed away from the rest of the network so that no incoming request can get directly to a private network resource. If configured properly, systems on one side of the firewall are protected from systems on the other side of the firewall.

- A firewall is an intrusion detection mechanism. Firewalls are specific to an organization's security policy. The settings of the firewalls can be changed to make appropriate changes to the firewall functionality.
- Firewalls can be configured to restrict incoming traffic to POP and SNMP and to enable email access. Certain firewalls block the email services to secure against spam.
- Firewalls can be configured to check inbound traffic at a point called the "choke point," where security audit is performed. The firewall can also act as an active "phone tap" tool in identifying the intruder's attempt to dial into the modems within the network

that is secured by firewall. The firewall logs consist of logging information that reports to the administrator on all the attempts of various incoming services.

- The firewall verifies the incoming and outgoing traffic against **firewall rules**. It acts as a router to move data between networks. Firewalls manage access of private networks to host applications.
- All the attempts to log in to the network are identified for auditing. Unauthorized attempts can be identified by embedding an alarm that is triggered when an unauthorized user attempts to login. Firewalls can filter packets based on address and types of traffic. They identify the source, destination addresses, and port numbers while address filtering, and they identify types of network traffic when protocol filtering. Firewalls can identify the state and attributes of the data packets.

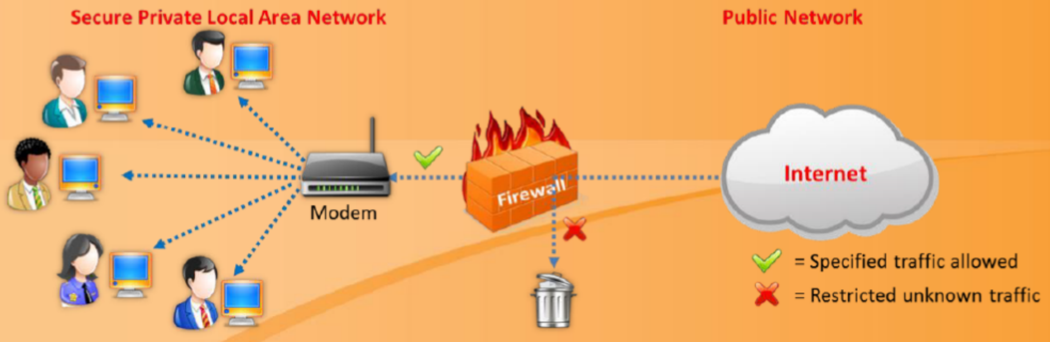
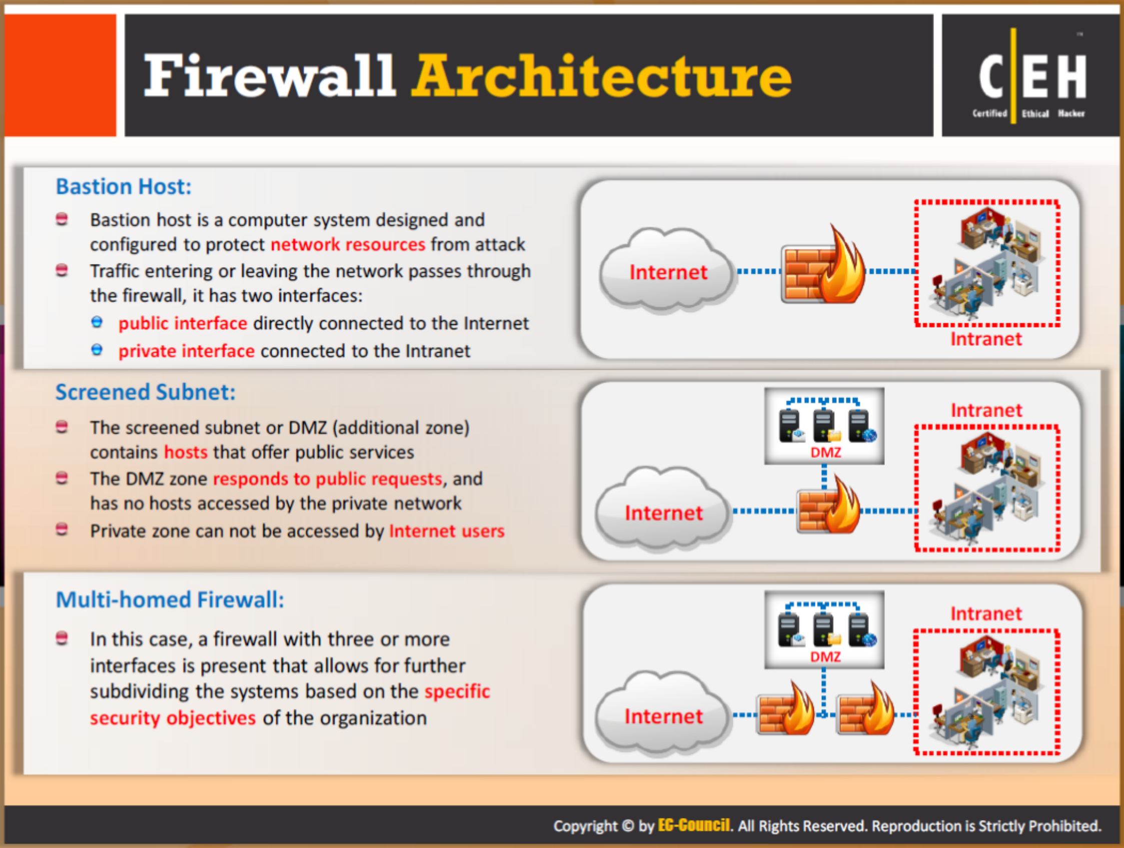

Secure Private Local Area Network
Public Network
Modem
Firewall
Internet
✓ = Specified traffic allowed
✗ = Restricted unknown traffic

FIGURE 17.4: Working of Firewall



Firewall Architecture

Bastion Host:

- Bastion host is a computer system designed and configured to protect network resources from attack
- Traffic entering or leaving the network passes through the firewall, it has two interfaces:
  - public interface directly connected to the Internet
  - private interface connected to the Intranet

Internet
Intranet

Screened Subnet:

- The screened subnet or DMZ (additional zone) contains hosts that offer public services
- The DMZ zone responds to public requests, and has no hosts accessed by the private network
- Private zone can not be accessed by Internet users

Multi-homed Firewall:

- In this case, a firewall with three or more interfaces is present that allows for further subdividing the systems based on the specific security objectives of the organization

Firewall Architecture

Firewall architecture consists of the following elements:

Bastion host

The bastion host is designed for the purpose of **defending against attacks**. It acts as a mediator between inside and outside networks. A bastion host is a computer system designed and configured to protect **network resources from attack**.

Traffic entering or leaving the network passes through the firewall, it has two interfaces:

- Public interface directly connected to the Internet
- Private interface connected to the intranet

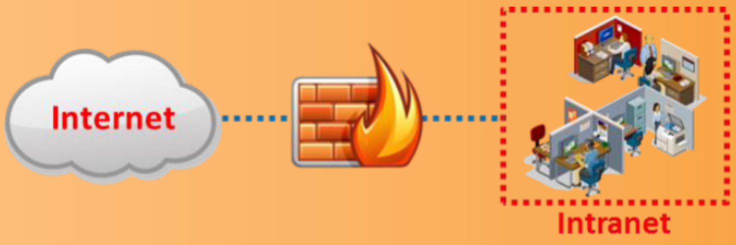


FIGURE 17.5: Bastion Host Architecture

Screened subnet

A screened subnet is a network architecture that uses a **single firewall** with three network interfaces. The first interface is used to connect the Internet, the second interface is used to connect the **DMZ**, the third interface is used to connect the intranet.

The main advantage with the screened subnet is it separates the DMZ and Internet from the intranet so that when the firewall is compromised access to the intranet won't be possible.

- The screened subnet or **DMZ (additional zone)** contains hosts that offer public services
- Public zone is directly connected to the Internet and has no hosts controlled by the organization
- Private zone has systems that Internet users have no business accessing

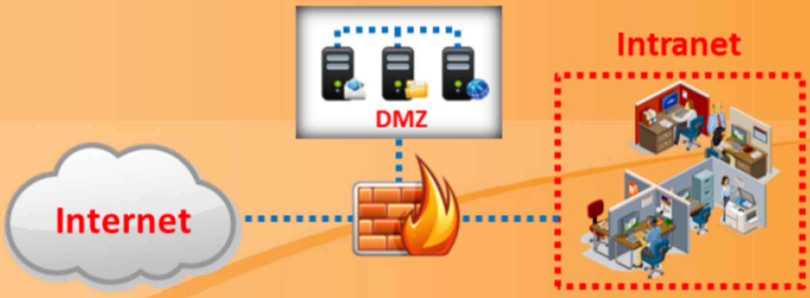


FIGURE 17.6: Screened Subnet Architecture

Multi-homed firewall

A multi-homed firewall generally refers to **two are more networks**. Each interface is connected to the **separate network segments** logically and physically. A multi-homed firewall is used to increase efficiency and reliability of an IP network. In this case, more than three interfaces are present that allow for further subdividing the systems based on the specific security objectives of the organization.

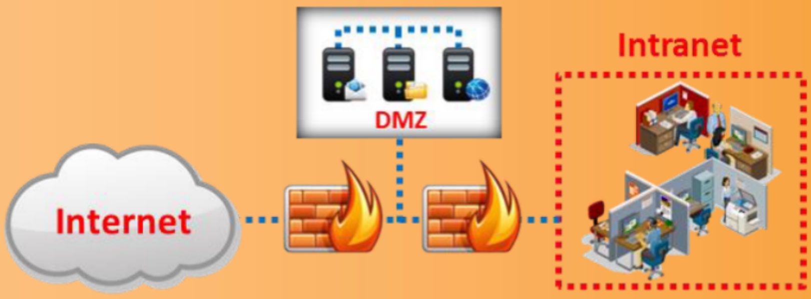
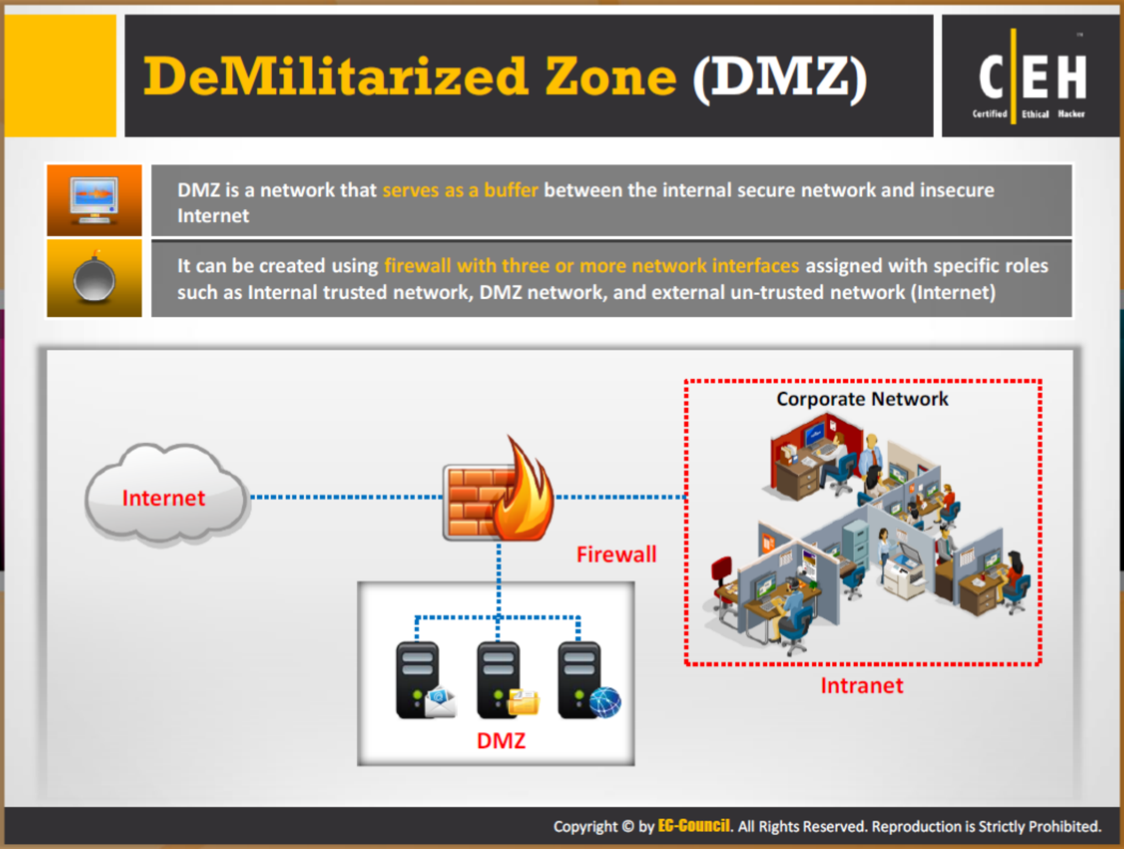


FIGURE 17.7: Multi-Homed Firewall Architecture



DeMilitarized Zone (DMZ)

- DMZ is a network that serves as a buffer between the internal secure network and insecure Internet
- It can be created using firewall with three or more network interfaces assigned with specific roles such as Internal trusted network, DMZ network, and external un-trusted network (Internet)

Demilitarized Zone (DMZ)

The DMZ is a **host computer** or a **network** placed as a neutral network between a particular firm's internal, or private, network and outside, or public, network to prevent the outside user from accessing the company's private data. DMZ is a network that serves as a buffer between the **internal secure network** and **insecure internet**

It is created using a firewall with three or more network interfaces assigned with specific roles such as Internal trusted network, DMZ network, and External un-trusted network (Internet).

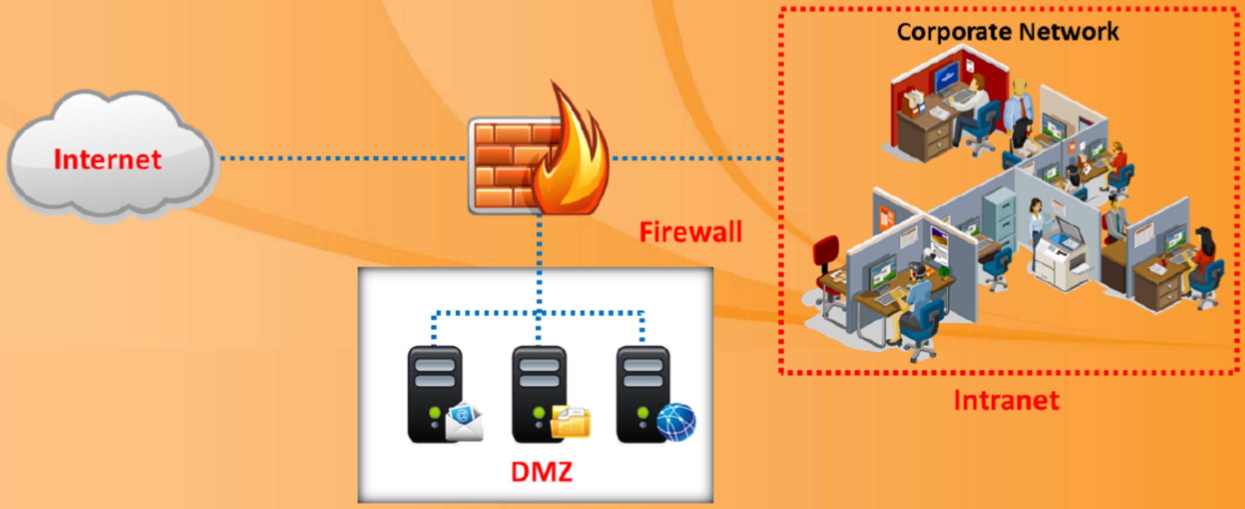
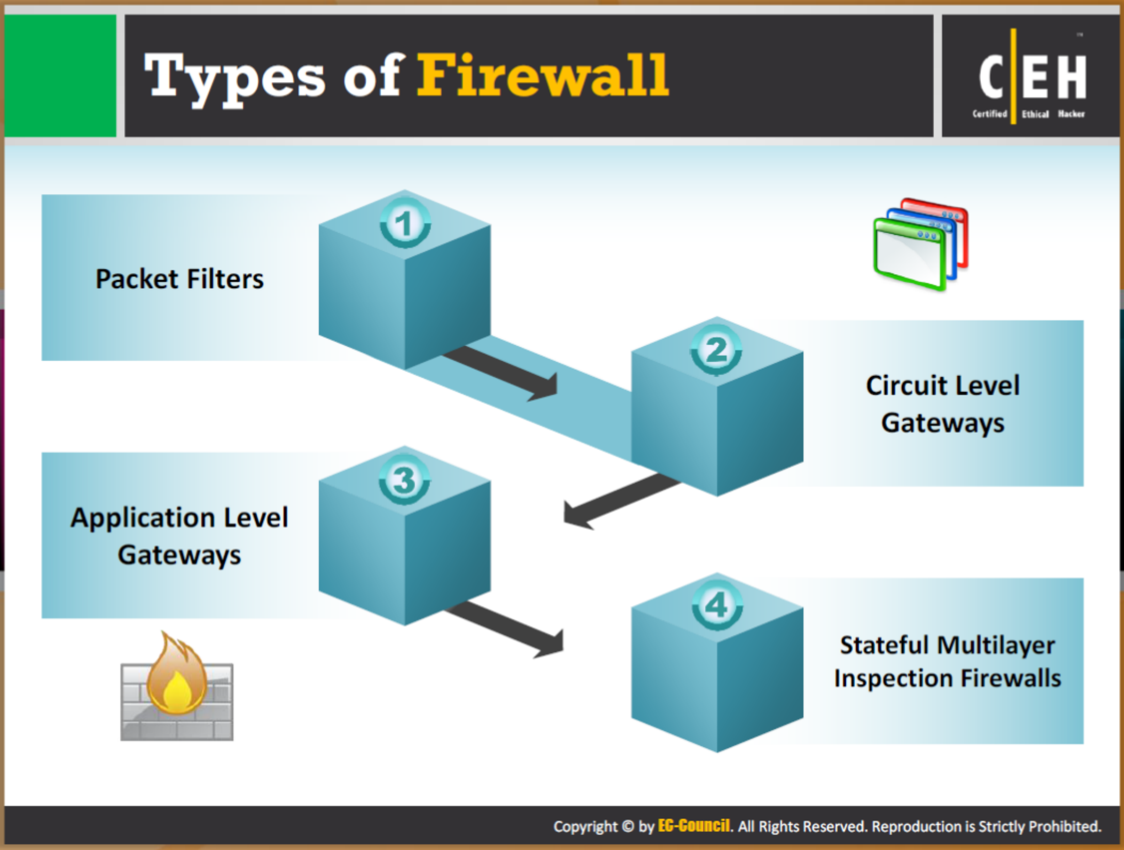

Corporate Network
Internet
Firewall
DMZ
Intranet

FIGURE 17.8: Demilitarized Zone (DMZ)



Types of Firewall

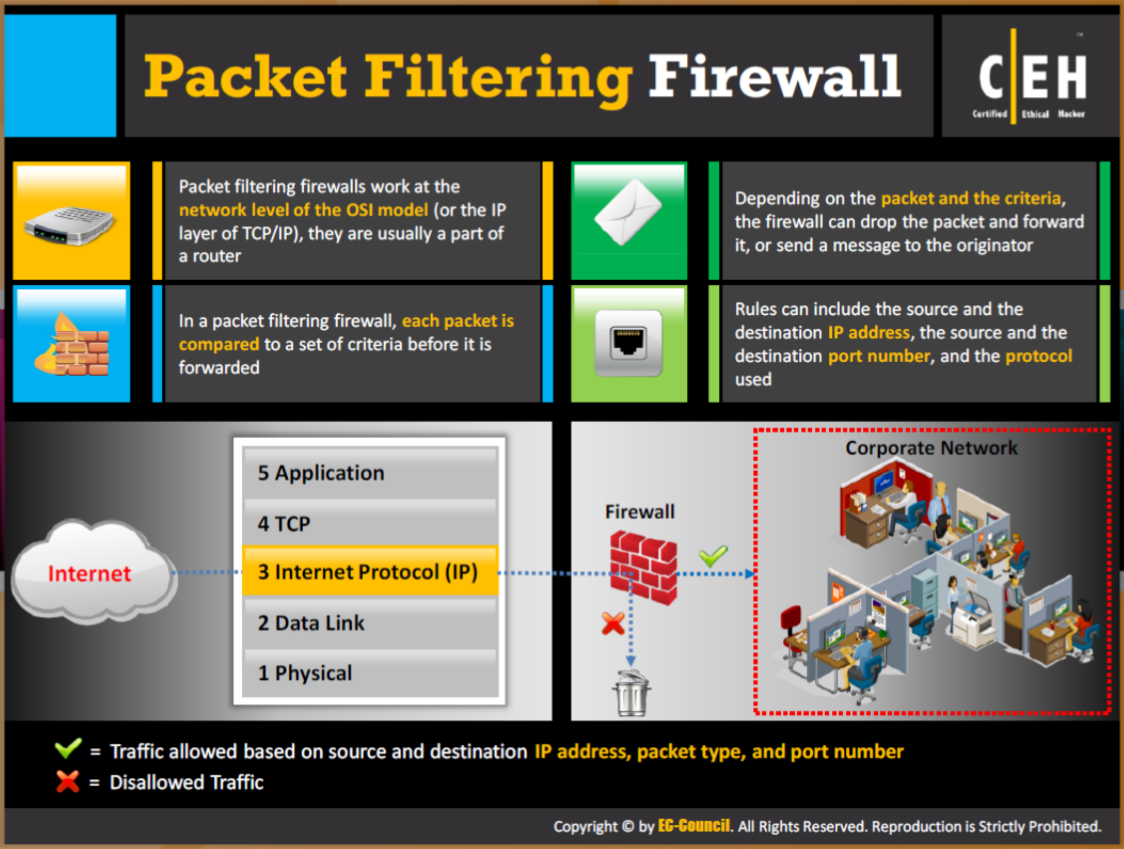
1
Packet Filters
2
Circuit Level Gateways
3
Application Level Gateways
4
Stateful Multilayer Inspection Firewalls

Types of Firewalls

A firewall refers to a **hardware device** or a **software program** used in a system to prevent malicious information from passing through and allowing only the approved information.

Firewalls are mainly categorized into four types:

- Packet filters
- Circuit-level gateways
- Application-level gateways
- Stateful multilayer inspection firewalls



Packet Filtering Firewall

Packet filtering firewalls work at the **network level of the OSI model** (or the IP layer of TCP/IP), they are usually a part of a router

Depending on the **packet and the criteria**, the firewall can drop the packet and forward it, or send a message to the originator

In a packet filtering firewall, **each packet is compared** to a set of criteria before it is forwarded

Rules can include the source and the destination **IP address**, the source and the destination **port number**, and the **protocol** used

✓ = Traffic allowed based on source and destination **IP address, packet type, and port number**

✗ = Disallowed Traffic

Packet Filtering Firewall

A packet filtering firewall investigates each **individual packet** passing through it and makes a decision whether to pass the packet or drop it. As you can tell from their name, packet **filter-based firewalls** concentrate on individual packets and analyze their header information and which way they are directed.

Traditional packet filters make the decision based on the following information:

- **Source IP address:** This is used to check if the packet is coming from a valid source or not. The information about the source IP address can be found from the **IP header** of the packet, which indicates the source system address.
- **Destination IP address:** This is used to check if the packet is going to the correct destination and to check if the destination accepts these types of packets. The information about the destination IP address can be found from the IP header of the packet, which has the destination address.
- **Source TCP/UDP port:** This is used to check the **source port** for the packet.
- **Destination TCP/UDP port:** This is used to check the destination port for the services to be allowed and the **services to be denied**.

- **TCP code bits:** Used to check whether the packet has a **SYN**, **ACK**, or other bits set for the connection to be made.
- **Protocol in use:** Used to check whether the protocol that the packet is carrying should be allowed. This is because some networks do not allow the **UDP protocol**.
- **Direction:** Used to check whether the packet is coming from the packet filter firewall or leaving it.
- **Interface:** Used to check whether or not the packet is coming from an unreliable site.

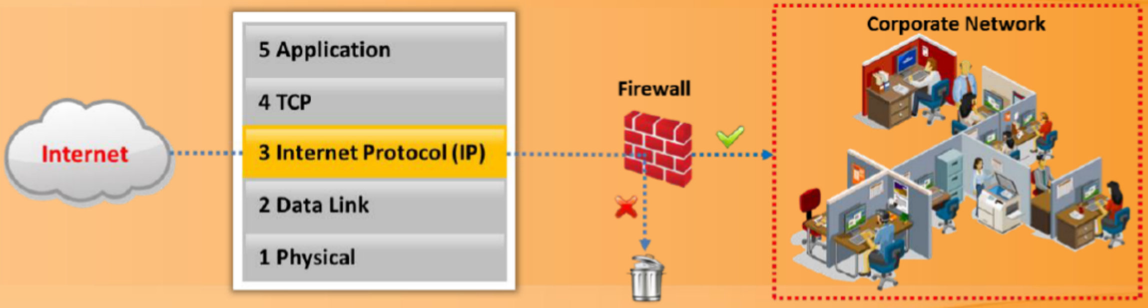
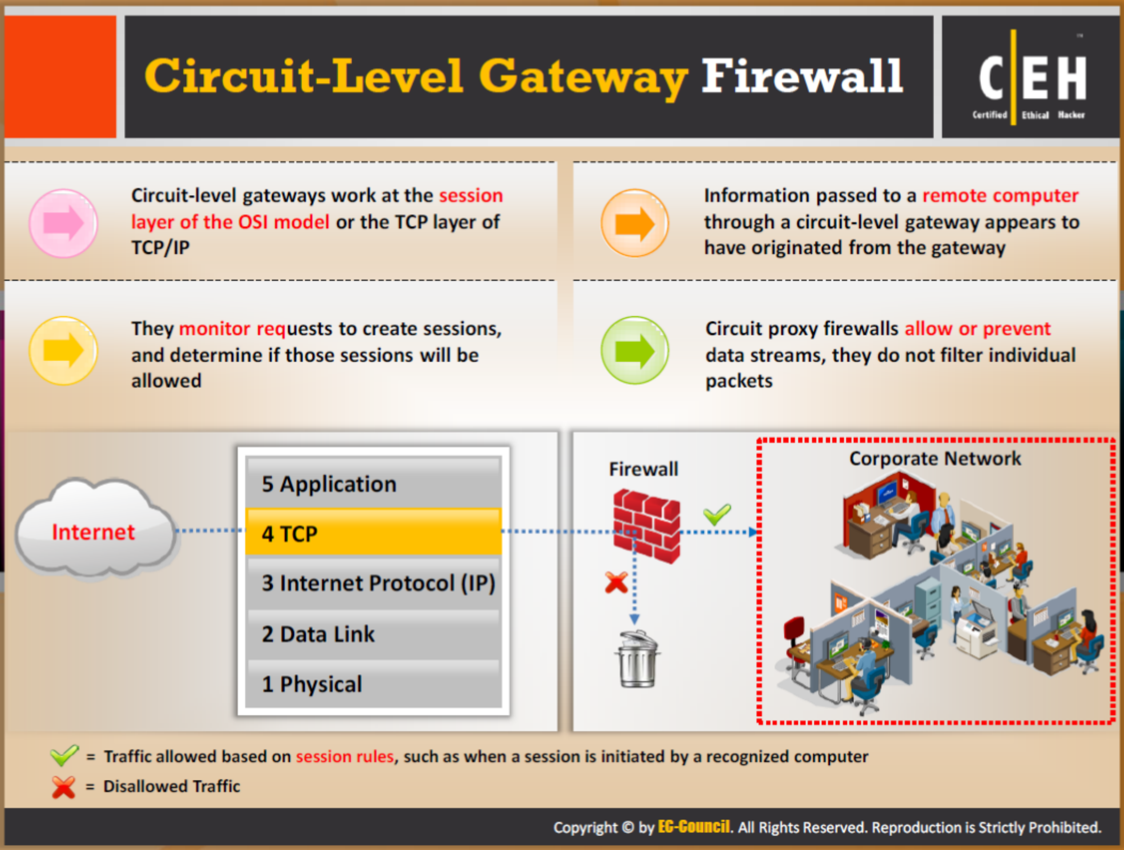

Internet
5 Application
4 TCP
3 Internet Protocol (IP)
2 Data Link
1 Physical
Firewall
Corporate Network

FIGURE 17.9: Packet Filtering Firewall

✓ = Traffic allowed based on source and destination **IP address, packet type, and port number**
✗ = Disallowed Traffic



Circuit-Level Gateway Firewall

- Circuit-level gateways work at the session layer of the OSI model or the TCP layer of TCP/IP
- Information passed to a remote computer through a circuit-level gateway appears to have originated from the gateway
- They monitor requests to create sessions, and determine if those sessions will be allowed
- Circuit proxy firewalls allow or prevent data streams, they do not filter individual packets

✓ = Traffic allowed based on session rules, such as when a session is initiated by a recognized computer

✗ = Disallowed Traffic

Circuit-level Gateway Firewall

Circuit-level gateways work at the session layer of the **OSI model** or the **TCP layer** of **TCP/IP**. A circuit-level gateway forwards data between the networks without verifying it. It blocks incoming packets into the host, but allows the traffic to pass through itself. Information passed to remote computers through a circuit-level gateway appears to have originated from the gateway, as the incoming traffic carries the **IP address** of the proxy (**circuit-level gateway**).

A circuit-level gateway gives the controlled network connection to the network between the system, internal and external to it. For detecting whether or not a requested session is valid, it checks the **TCP handshaking** between the packets. Circuit-level gateways do not filter individual packets. Circuit-level gateways are relatively inexpensive and hide the information about the private network that they protect.

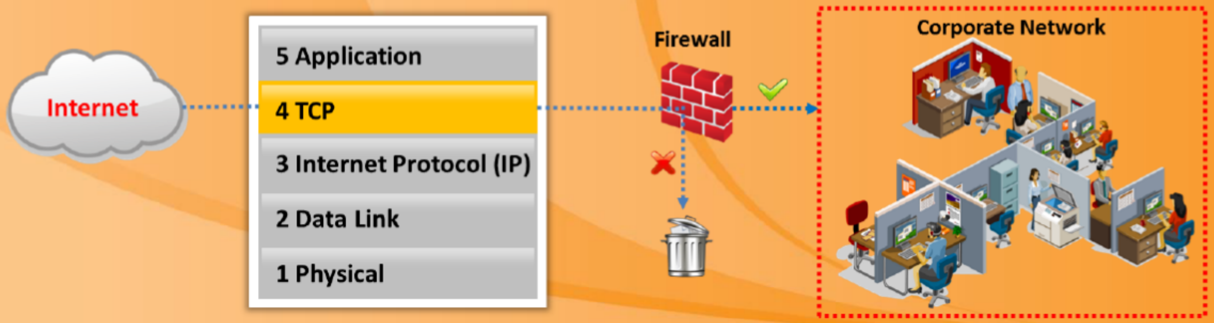
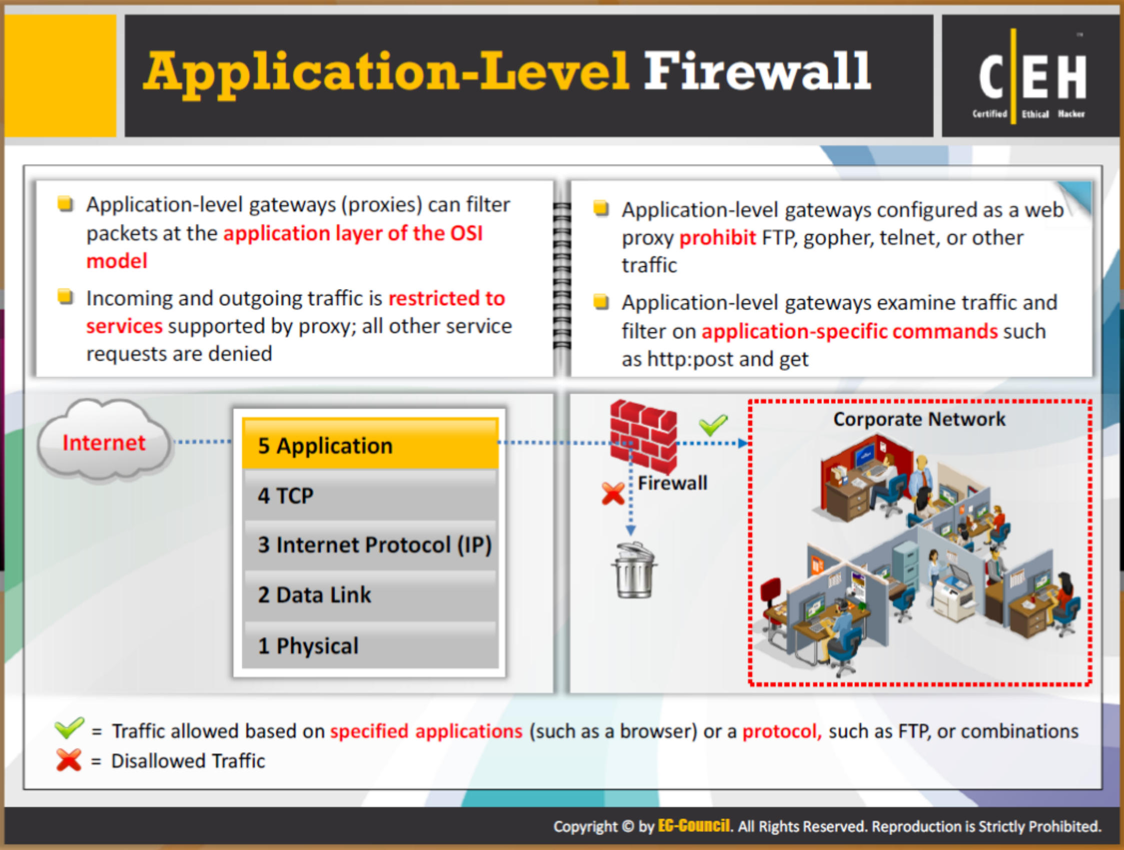

Internet
5 Application
4 TCP
3 Internet Protocol (IP)
2 Data Link
1 Physical
Firewall
Corporate Network

FIGURE 17.10: Circuit-level Gateway Firewall

✓ = **Traffic allowed based on session rules, such as when a session is initiated by a recognized computer**

✗ = **Disallowed Traffic**



Application-Level Firewall

- Application-level gateways (proxies) can filter packets at the **application layer of the OSI model**
- Incoming and outgoing traffic is **restricted to services** supported by proxy; all other service requests are denied
- Application-level gateways configured as a web proxy **prohibit** FTP, gopher, telnet, or other traffic
- Application-level gateways examine traffic and filter on **application-specific commands** such as http:post and get

Internet

5 Application
4 TCP
3 Internet Protocol (IP)
2 Data Link
1 Physical

Firewall

Corporate Network

✓ = Traffic allowed based on **specified applications** (such as a browser) or a **protocol,** such as FTP, or combinations
✗ = Disallowed Traffic

Application-level Firewall

Proxy/application-based firewalls concentrate on the Application layer rather than just the packets.

- These firewalls analyze the application information to make decisions about whether or not to transmit the packets.
- A **proxy-based** firewall asks for authentication to pass the packets as it works at the Application layer.
- A content caching proxy optimizes performance by caching frequently accessed information instead of sending new requests for the same old data to the servers.

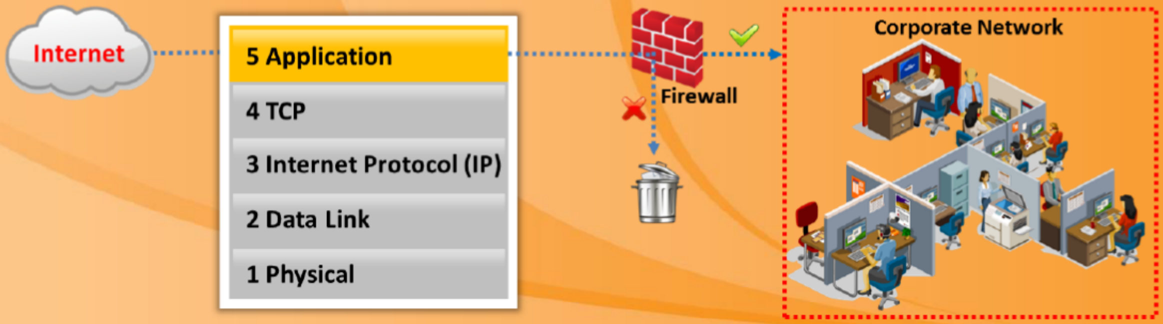
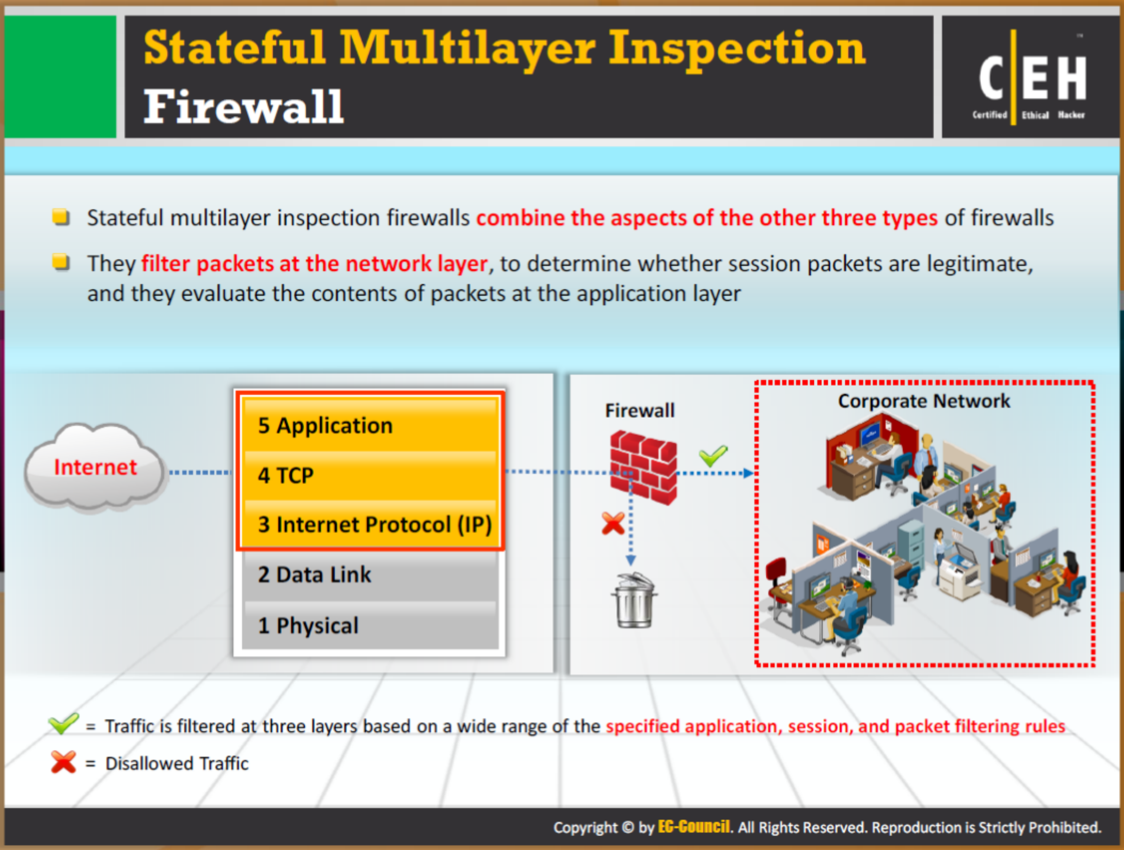

Internet
5 Application
4 TCP
3 Internet Protocol (IP)
2 Data Link
1 Physical
Firewall
Corporate Network

FIGURE 17.11: Application-level Firewall



Stateful Multilayer Inspection Firewall

CEH Certified Ethical Hacker

- Stateful multilayer inspection firewalls **combine the aspects of the other three types** of firewalls
- They **filter packets at the network layer**, to determine whether session packets are legitimate, and they evaluate the contents of packets at the application layer

✓ = Traffic is filtered at three layers based on a wide range of the **specified application, session, and packet filtering rules**

✗ = Disallowed Traffic

Stateful Multilayer Inspection Firewall

Stateful multilayer inspection firewalls combine the aspects of the other **three types of firewalls**. They filter packets at the network layer, to determine whether session packets are legitimate, and they evaluate the contents of packets at the application layer.

The inability of the packet filter firewall to check the header of the packets to allow the passing of packets is overcome by stateful packet filtering.

- This type of firewall can remember the packets that passed through it earlier and make decisions about future packets based on memory
- These firewalls provide the best of both **packet filtering** and **application-based filtering**
- **Cisco PIX** firewalls are stateful
- These firewalls tracks and log slots or translations

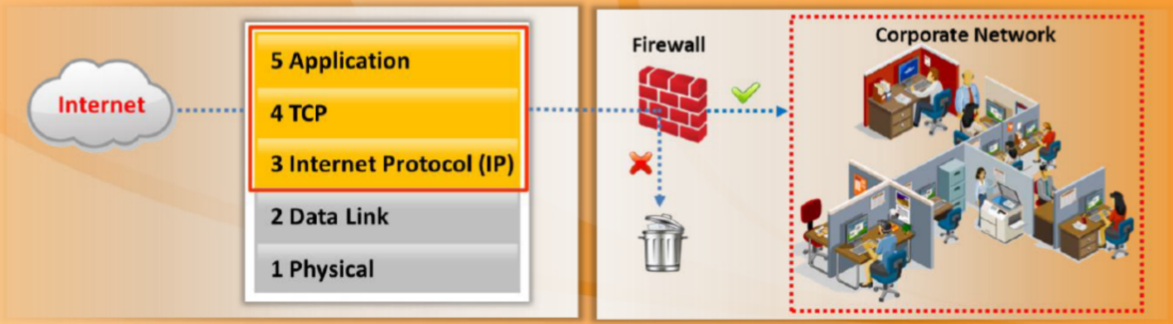
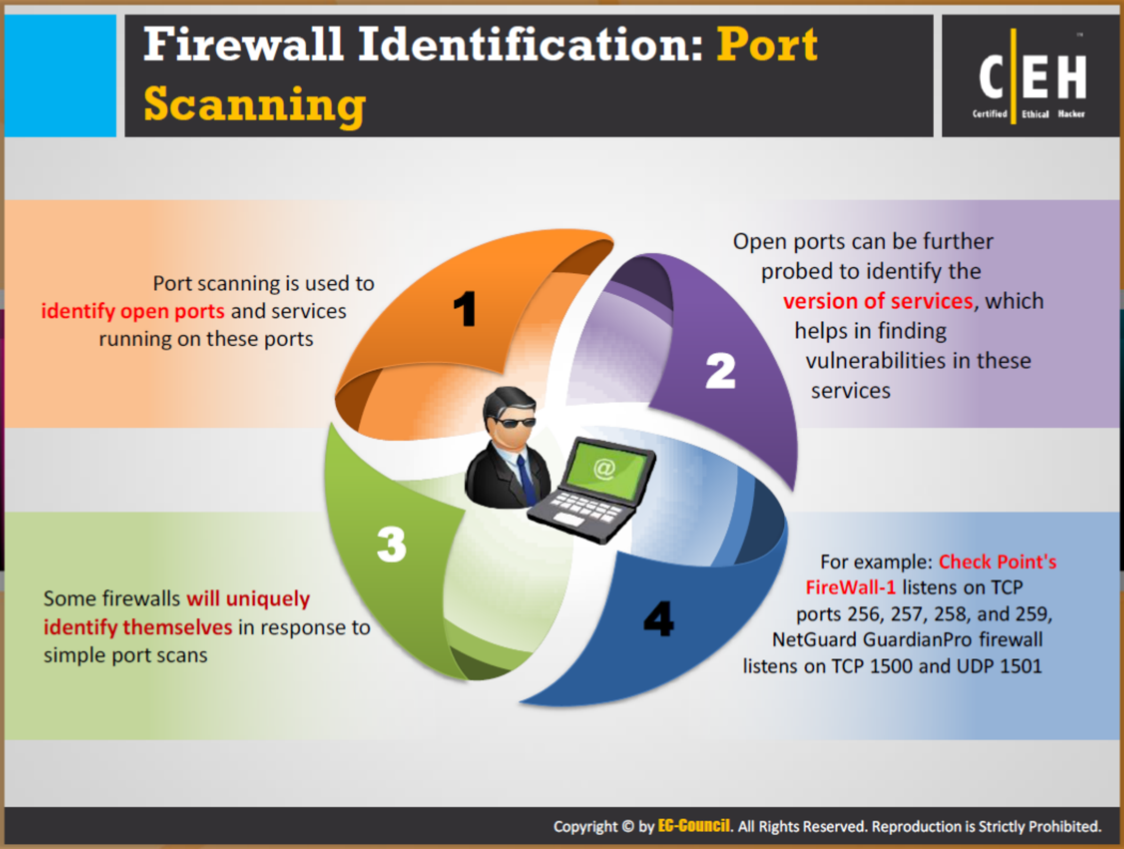

Internet
5 Application
4 TCP
3 Internet Protocol (IP)
2 Data Link
1 Physical
Firewall
Corporate Network

FIGURE 17.12: Stateful Multilayer Inspection Firewall

✓ = Traffic is filtered at three layers based on a wide range of the **specified application, session, and packet filtering rules**

✗ = Disallowed Traffic



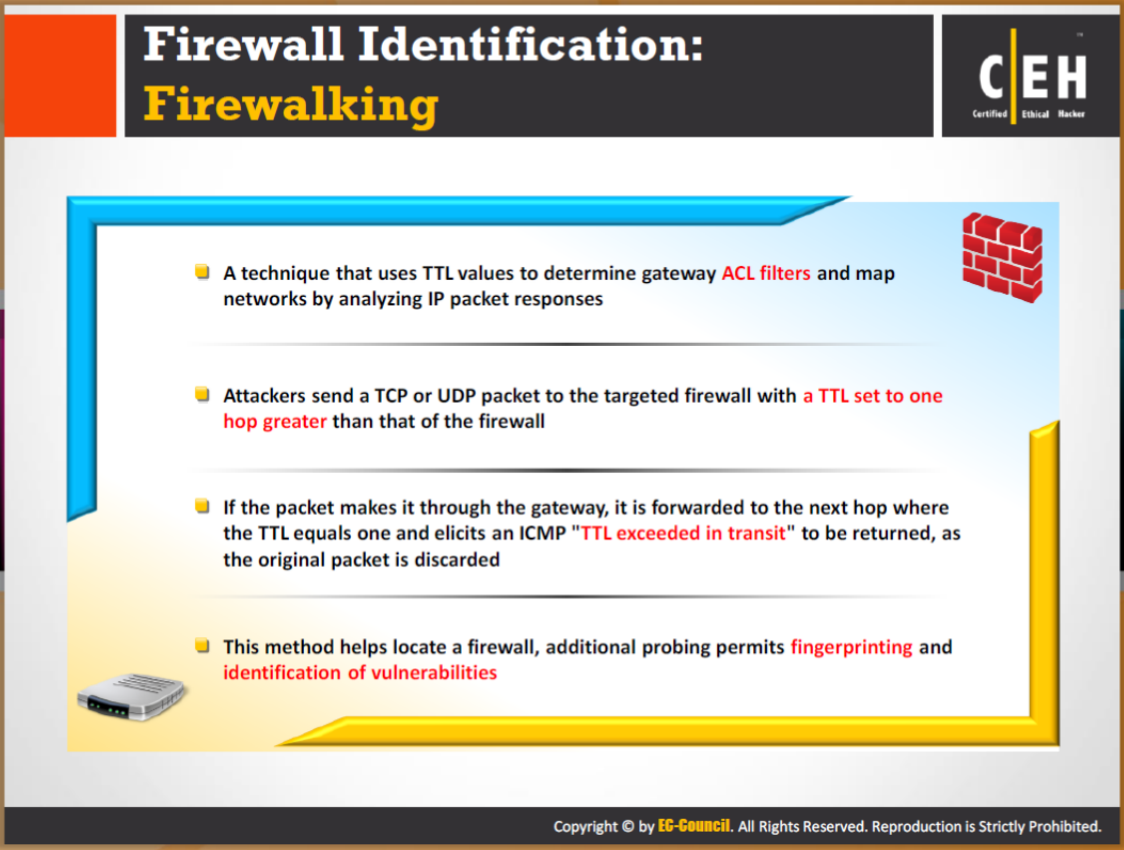
Firewall Identification: Port Scanning

1. Port scanning is used to identify open ports and services running on these ports
2. Open ports can be further probed to identify the version of services, which helps in finding vulnerabilities in these services
3. Some firewalls will uniquely identify themselves in response to simple port scans
4. For example: Check Point's FireWall-1 listens on TCP ports 256, 257, 258, and 259, NetGuard GuardianPro firewall listens on TCP 1500 and UDP 1501

Firewall Identification: Port Scanning

Systematically scanning the ports of a computer is known as port scanning. Attackers use such methods to identify the possible vulnerabilities in order to compromise a network. It is one of the most popular methods that attackers use for investigating the ports used by the victims. A tool that can be used for port scanning is Nmap.

A port scan helps the attacker find which ports are available (i.e., what service might be listening to a port); it consists of sending a message to each port, one at a time. The kind of response received indicates whether the port is used and can therefore be probed further for weakness. Some firewalls will uniquely identify themselves using simple port scans. For example: Check Point's FireWall-1 listens on TCP ports 256, 257, 258, and 259 and Microsoft's Proxy Server usually listens on TCP ports 1080 and 1745.



Firewall Identification: Firewalking

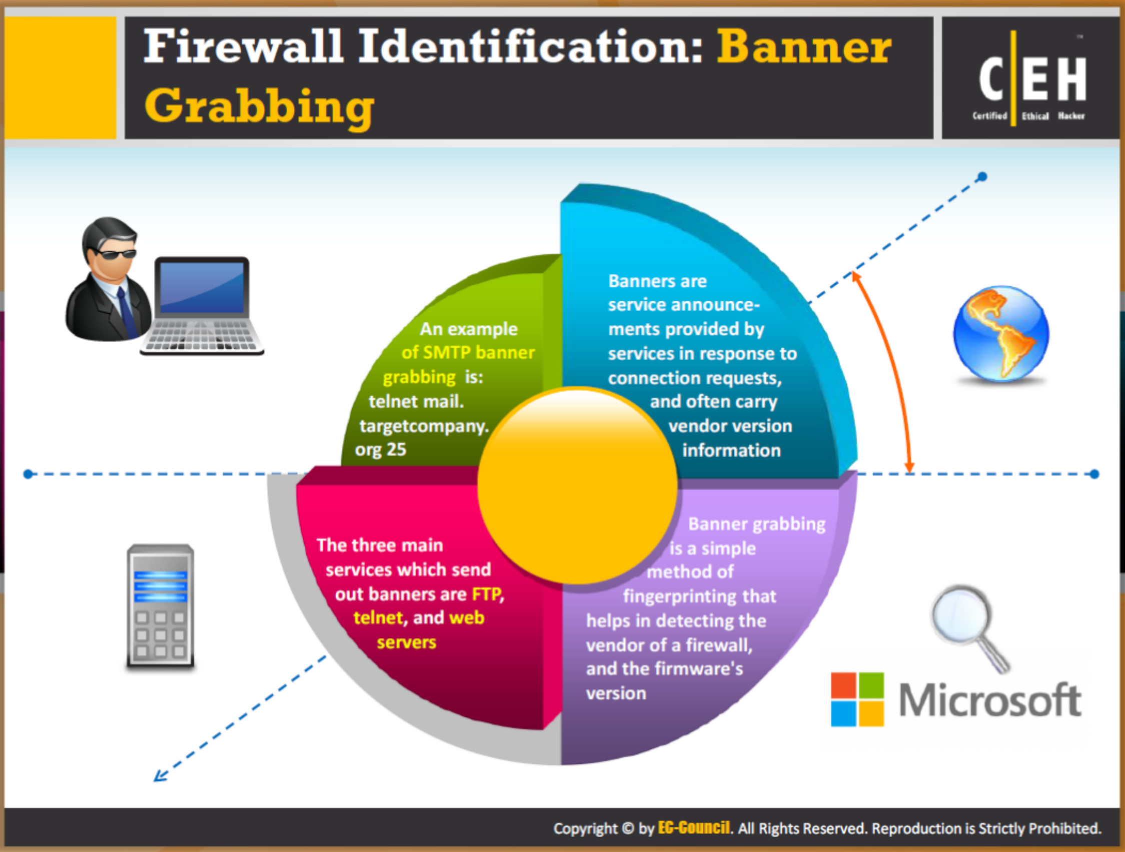
- A technique that uses TTL values to determine gateway ACL filters and map networks by analyzing IP packet responses
- Attackers send a TCP or UDP packet to the targeted firewall with a TTL set to one hop greater than that of the firewall
- If the packet makes it through the gateway, it is forwarded to the next hop where the TTL equals one and elicits an ICMP "TTL exceeded in transit" to be returned, as the original packet is discarded
- This method helps locate a firewall, additional probing permits fingerprinting and identification of vulnerabilities

Firewall Identification: Firewalking

Firewalking is a method used to collect information about remote networks that are behind firewalls. It probes ACLs on packet filtering **routers/firewalls**. It is same as that of tracerouting and works by sending TCP or UDP packets into the firewall that have a TTL set at one hop greater than the targeted firewall. If the packet makes it through the gateway, it is forwarded to the next hop where the TTL equals zero and elicits a TTL "exceeded in transit" message, at which point the packet is discarded. Using this method, access information on the firewall can be determined if successive probe packets are sent.

Firewalk is the most well-known software used for firewalking. It has two phases: **a network discovery phase** and a **scanning phase**. It requires three hosts:

- **Firewalking host:** The firewalking host is the system, outside the target network, from which the data packets are sent, to the destination host, in order to gain more information about the target network.
- **Gateway host:** The gateway host is the system on the target network that is connected to the Internet, through which the data packet passes on its way to the target network.
- **Destination host:** The destination host is the target system on the target network that the data packets are addressed to.



Firewall Identification: Banner Grabbing

An example of SMTP banner grabbing is: telnet mail.targetcompany.org 25

Banners are service announcements provided by services in response to connection requests, and often carry vendor version information

The three main services which send out banners are FTP, telnet, and web servers

Banner grabbing is a simple method of fingerprinting that helps in detecting the vendor of a firewall, and the firmware's version

Firewall Identification: Banner Grabbing

Banners are messages sent out by **network services** during the connection to the service. Banners announce which service is running on the system. Banner grabbing is a technique generally used by the attacker for **OS detection**. The attacker uses banner grabbing to discover services run by **firewalls**. The three main services that send out banners are FTP, Telnet, and web servers.

Ports of services such as FTP, Telnet, and web servers should not be kept open, as they are vulnerable to banner grabbing. A firewall does not **block banner grabbing** because the connection between the attacker's system and the target system looks legitimate.

An example of SMTP banner grabbing is: telnet mail.targetcompany.org 25. The syntax is: `"<service name > <service running > <port number>"`

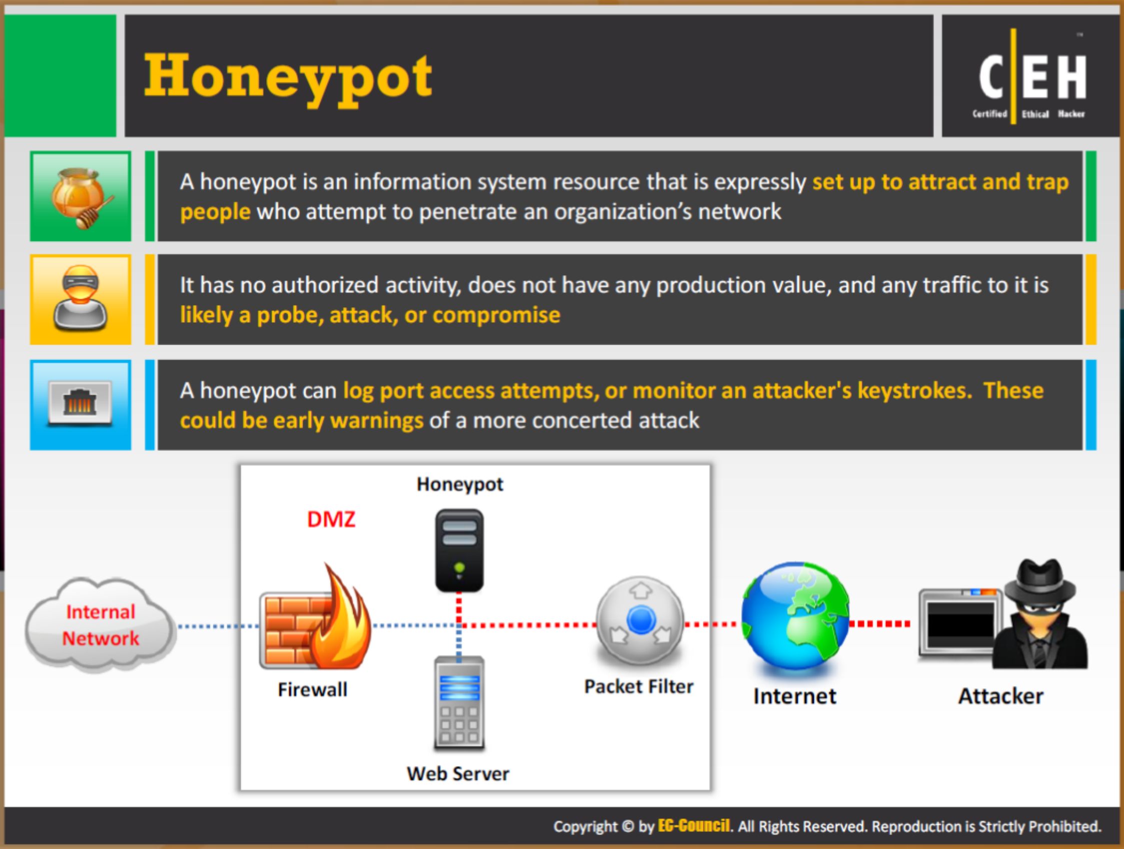
Banner grabbing is a mechanism that is tried and true for specifying banners and application information. For example, when the user opens a telnet connection to a known port on the target server and presses Enter a few times, if required, the following result is displayed:

```
C:\>telnet www.corleone.com 80

HTTP/1.0 400 Bad Request

Server: Netscape – Commerce/1.12
```

This system works with many other common applications that respond on a set port. The information generated through banner grabbing can enhance the attacker's efforts to further compromise the system. With information about the version and the vendor of the web server, the attacker can further concentrate on employing platform-specific exploit techniques.



Honeypot

- A honeypot is an information system resource that is expressly **set up to attract and trap people** who attempt to penetrate an organization's network
- It has no authorized activity, does not have any production value, and any traffic to it is **likely a probe, attack, or compromise**
- A honeypot can **log port access attempts, or monitor an attacker's keystrokes. These could be early warnings** of a more concerted attack

Honeypot

A honeypot is a system that is intended to **attract** and **trap** people who try unauthorized or **illicit utilization of the host system**. Whenever there is any interaction with a honeypot, it is most likely to be a malicious activity. Honeypots are unique; they do not solve a specific problem. Instead, they are a **highly flexible tool** with many different security applications. Some honeypots can be used to help prevent attacks; others can be used to detect attacks; while a few honeypots can be used for information gathering and research.

Examples:

- Installing a system on the network with no particular purpose other than to log all attempted access.
- Installing an older unpatched operating system on a network. For example, the default installation of WinNT 4 with IIS 4 can be hacked using several different techniques. A standard intrusion detection system can then be used to log hacks directed against the system and further track what the intruder attempts to do with the system once it is compromised. Install special software designed for this purpose. It has the advantage of making it look like the intruder is successful without really allowing him/her access to the network.

Any existing system can be "**honeypot-ized**." For example, on WinNT, it is possible to rename the default administrator account and then create a dummy account called "administrator" with no password. WinNT allows extensive logging of a person's activities, so this honeypot tracks users who are attempting to gain administrator access and exploit that access.

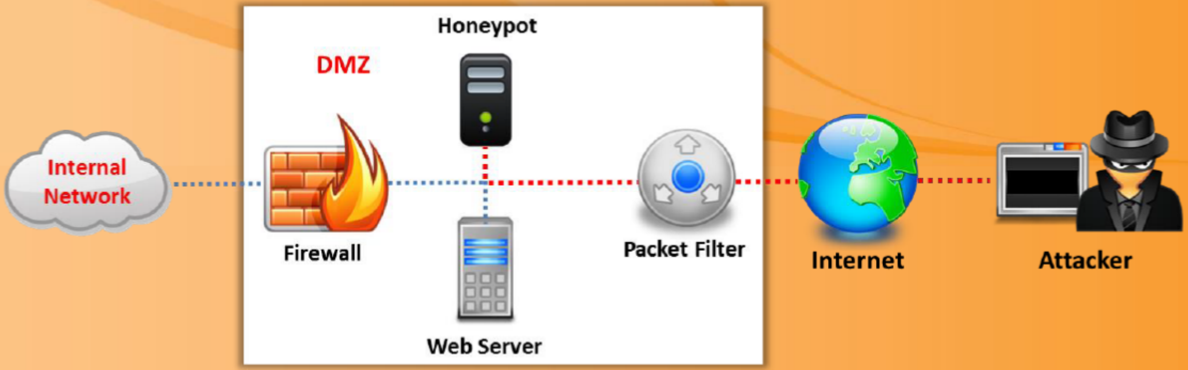
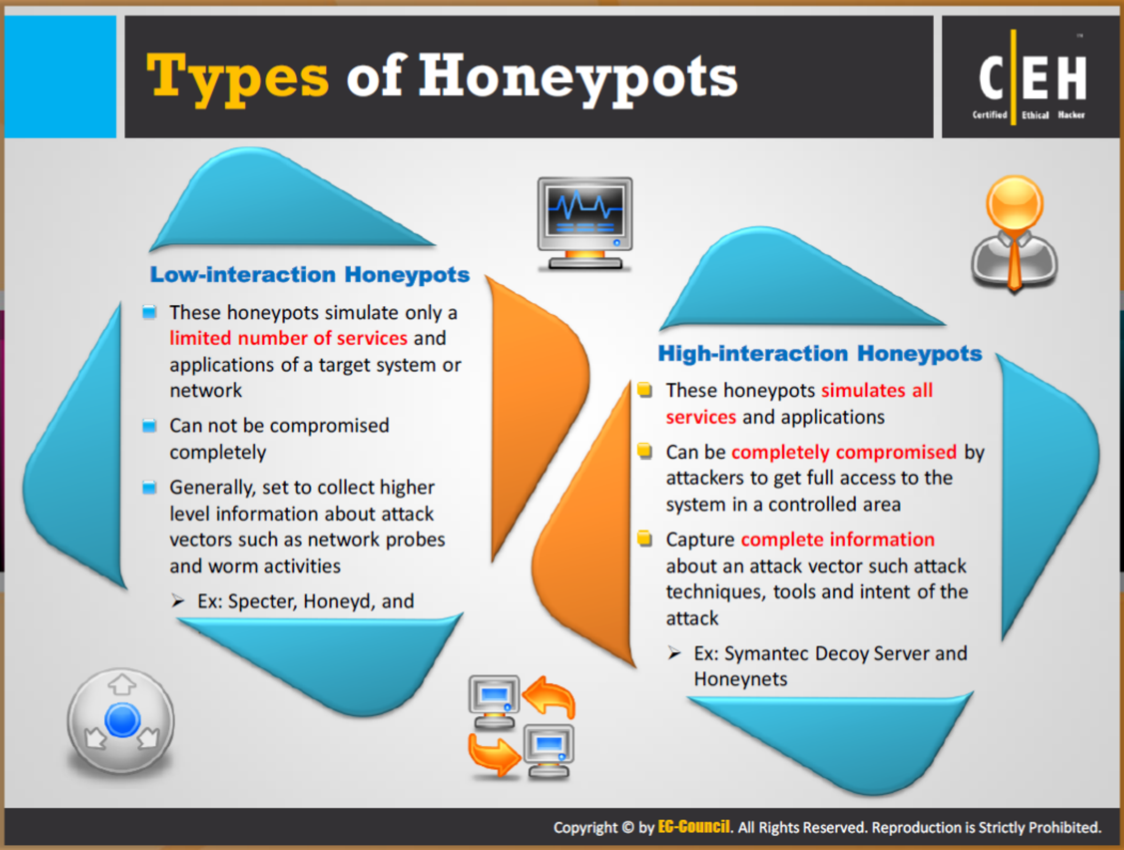

Honeypot
DMZ
Internal Network
Firewall
Packet Filter
Internet
Attacker
Web Server

FIGURE 17.13: Working of Honeypot



Types of Honeypots

Low-interaction Honeypots

- These honeypots simulate only a limited number of services and applications of a target system or network
- Can not be compromised completely
- Generally, set to collect higher level information about attack vectors such as network probes and worm activities
  - Ex: Specter, Honeyd, and

High-interaction Honeypots

- These honeypots simulates all services and applications
- Can be completely compromised by attackers to get full access to the system in a controlled area
- Capture complete information about an attack vector such attack techniques, tools and intent of the attack
  - Ex: Symantec Decoy Server and Honeynets

Types of Honeypots

Honeypots are mainly divided into two types:

Low-interaction Honeypot

They work by emulating services and programs that would be found on an individual's system. If the attacker does something that the emulation does not expect, the honeypot will simply generate an error. They capture limited amounts of information, mainly transactional data and some limited interaction

Ex: Specter, Honeyd, and KFSensor

Honeyd is a low-interaction honeypot. It is open source and designed to run primarily on UNIX systems. Honeyd works on the concept of monitoring unused IP space. Anytime it sees a connection attempt to an unused IP, it intercepts the connection and then interacts with the attacker, pretending to be the victim.

By default, Honeyd detects and logs connections to any UDP or TCP port. In addition, the user can configure emulated services to monitor specific ports, such as an emulated FTP server monitoring port 21 (TCP). When an attacker connects to the emulated service, not only does the honeypot detect and log the activity, but also it captures all of the attacker's interaction with the emulated service.

In the case of the emulated **FTP server**, an attacker's login and password can be potentially captured; the commands that were issued, what they were looking for, or their identity can be tracked. Most emulated services work the same way. They expect a specific type of behavior, and then are programmed to react in a predetermined way.

High-interaction Honeypot

Honeynets are a prime example of a **high-interaction honeypot**. A honeynet is neither a product nor a software solution that the user installs. Instead, it is architecture, an entire network of computers designed to attack.

The idea is to have an architecture that creates a **highly controlled network**, one where all activity is controlled and captured. Within this network, intended victims are placed and the network has real computers running real applications.

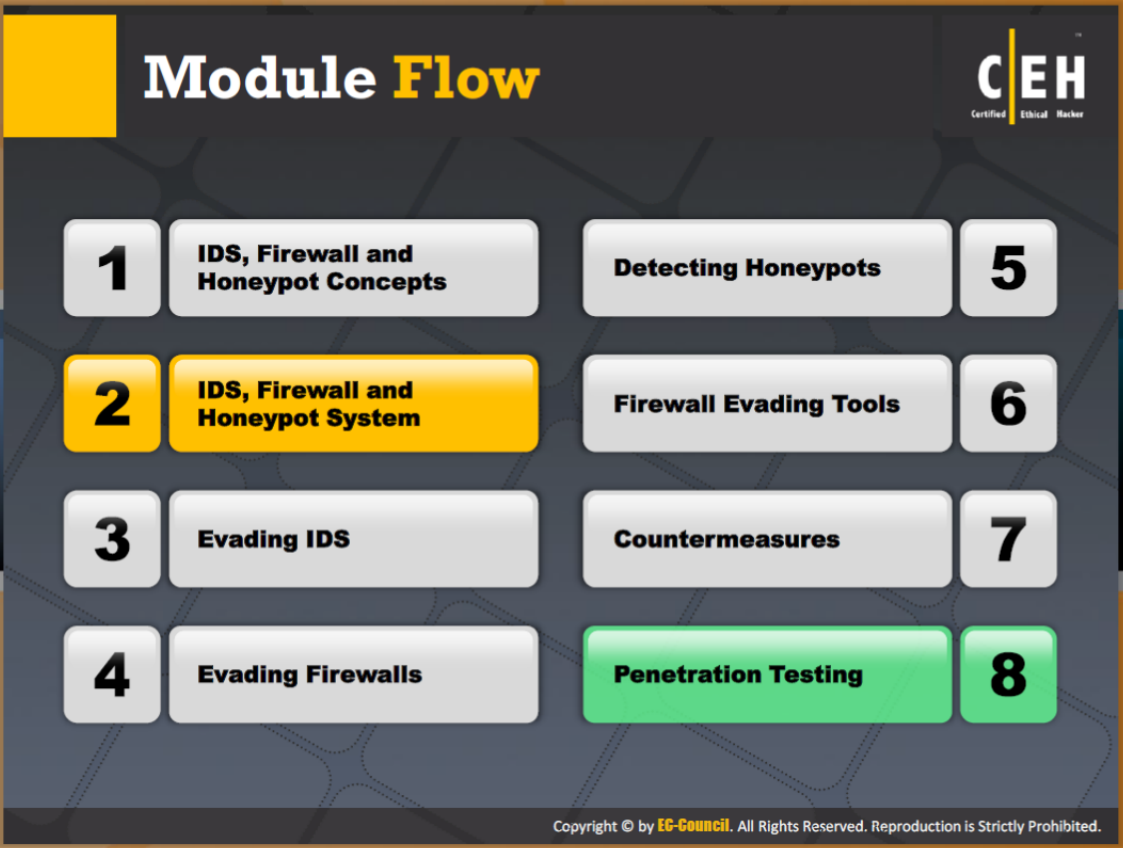
The "**bad guys**" find, attack, and break into these systems on their own initiative. When they do, they do not realize they are within a honeynet. All of their activity, from **encrypted SSH** sessions to email and file uploads, is captured without them knowing it by inserting kernel modules on the victim's systems, capturing all of the attacker's actions.

At the same time, the honeynet controls the attacker's activity. Honeynets do this by using a honeywall gateway. This gateway allows inbound traffic to the victim's systems, but controls the outbound traffic using intrusion prevention technologies. This gives the attacker the flexibility to interact with the victim's systems, but prevents the attacker from harming other non-honeynet computers.

How to Set Up a Honeypot

Follow the steps here to set up a honeypot:

- Step 1: Download or purchase **honeypot software**. Tiny Honeypot, LaBrea, and Honeyd are some of the programs available for Linux systems. KFSensor is software that works with Windows.
- Step 2: Log in as an administrator on the computer to install a honeypot onto the computer.
- Step 3: Install the software on your computer. Choose the "**Full Version**" to make sure every feature of the program is installed.
- Step 4: Place the honeypot software in the Program Files folder. Once you have chosen the folder, click"**OK** and the program will install.
- Step 5: Restart your computer for the honeypot to work.
- Step 6: Configure the honeypot to check the items that you want the honeypot to watch for, including services, applications, and Trojans, and name your domain.



Module Flow

1 IDS, Firewall and Honeypot Concepts
2 IDS, Firewall and Honeypot System
3 Evading IDS
4 Evading Firewalls
Detecting Honeypots 5
Firewall Evading Tools 6
Countermeasures 7
Penetration Testing 8

Module Flow

Previously, we discussed the basic concepts of three security mechanisms: IDSes, firewalls, and honeypots. Now we will move on to detailed descriptions and functionalities of these security mechanisms.

IDS, Firewall and Honeypot Concepts	Detecting Honeypots
IDS, Firewall and Honeypot System	Firewall Evading Tools
Evading IDS	Countermeasure
Evading Firewall	Penetration Testing

This section describes the intrusion detection system Snort.

Intrusion Detection Tool: Snort

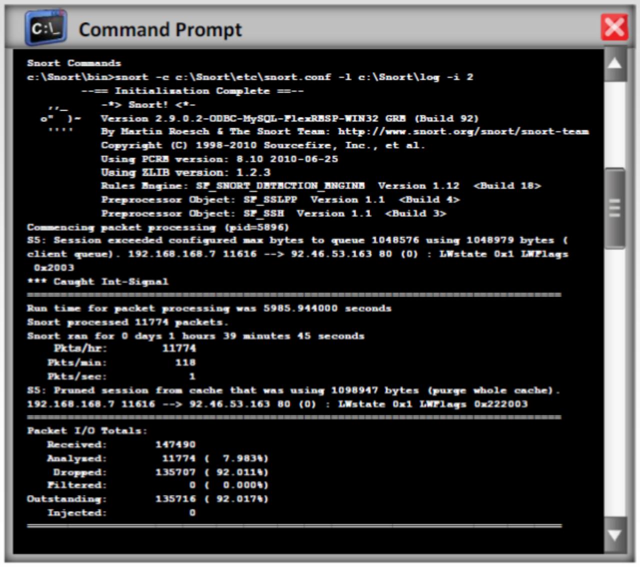
1 Snort is an open source network intrusion detection system, capable of performing real-time **traffic analysis and packet logging on IP networks**

2 It can perform **protocol analysis** and **content searching/matching**, and is used to detect a variety of **attacks and probes**, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, and OS fingerprinting attempts

3 It uses a flexible **rules language** to describe traffic that it should collect or pass, as well as a **detection engine** that utilizes a modular plug-in architecture

4 **Uses of Snort:**

- Straight packet sniffer like tcpdump
- Packet logger (useful for network traffic debugging, etc.)
- Network intrusion prevention system



```
Snort Commands
c:\Snort\bin>snort -c c:\Snort\etc\snort.conf -l c:\Snort\log -i 2
        --== Initialization Complete ==--
   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.0.2-ODBC-MySQL-FlexRESP-WIN32 GRE (Build 92)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team
           Copyright (C) 1998-2010 Sourcefire, Inc., et al.
           Using PCRE version: 8.10 2010-06-25
           Using ZLIB version: 1.2.3
           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 1.12  <Build 18>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
Commencing packet processing (pid=5896)
S5: Session exceeded configured max bytes to queue 1048576 using 1048979 bytes (
client queue). 192.168.168.7 11616 --> 92.46.53.163 80 (0) : LWstate 0x1 LWFlags
 0x2003
*** Caught Int-Signal
===============================================================================
Run time for packet processing was 5985.944000 seconds
Snort processed 11774 packets.
Snort ran for 0 days 1 hours 39 minutes 45 seconds
    Pkts/hr:        11774
   Pkts/min:          118
   Pkts/sec:            1
S5: Pruned session from cache that was using 1098947 bytes (purge whole cache).
192.168.168.7 11616 --> 92.46.53.163 80 (0) : LWstate 0x1 LWFlags 0x222003
===============================================================================
Packet I/O Totals:
   Received:       147490
   Analyzed:        11774 (  7.983%)
    Dropped:       135707 ( 92.011%)
   Filtered:            0 (  0.000%)
Outstanding:       135716 ( 92.017%)
   Injected:            0
===============================================================================
```

http://www.snort.org

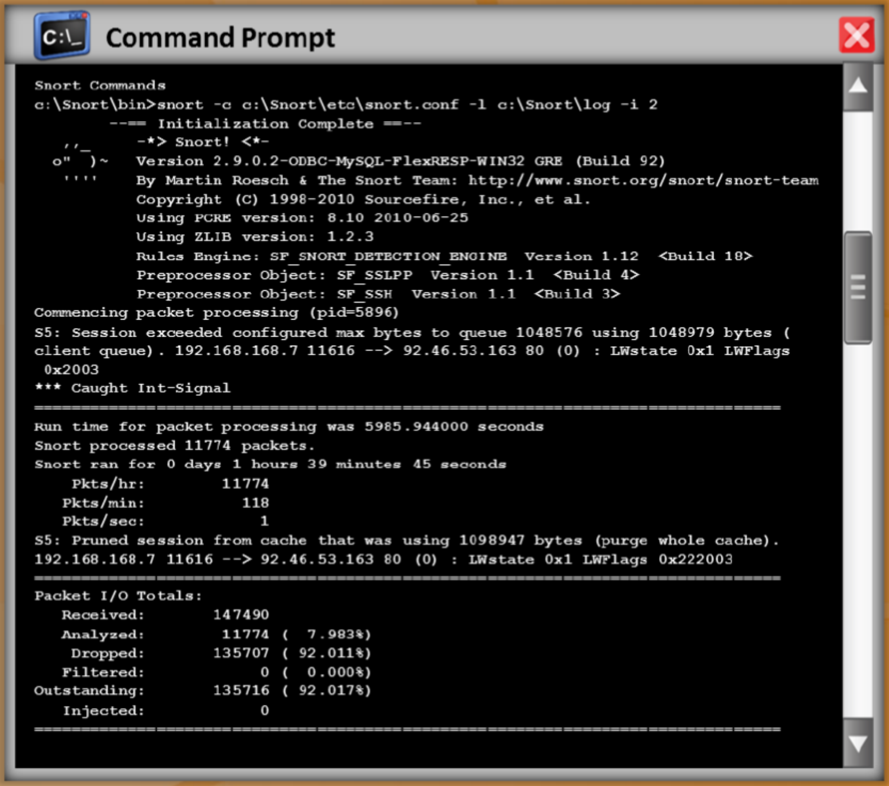
Intrusion Detection Tool: Snort

Source: http://www.snort.org

Snort is an open source network intrusion detection and prevention system capable of performing **real-time traffic analysis** and packet logging on **IP networks**. It can perform protocol analysis and content **searching/matching**. It can be used to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, OS fingerprinting, attempts etc.

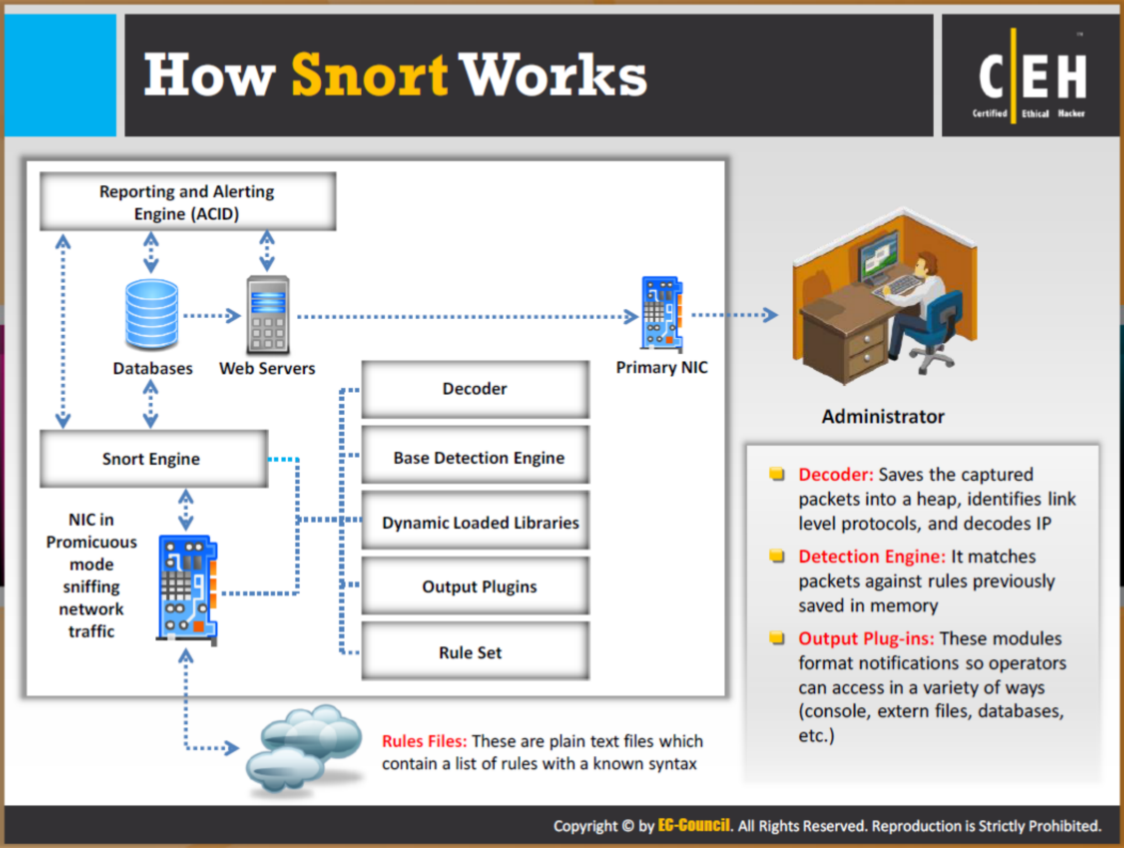
Snort uses a **flexible rules language** to describe traffic that it should collect or pass, as well as a detection engine that utilizes a modular **plug-in architecture**. Snort has a real-time alerting capability as well, incorporating alerting mechanisms for syslog, a user specified file, a UNIX socket, or WinPopup messages to Windows clients.

Snort has three primary uses: a straight packet sniffer like tcpdump, a packet logger (useful for network traffic debugging, etc.), or a full-blown network intrusion prevention system.



```
Snort Commands
c:\Snort\bin>snort -c c:\Snort\etc\snort.conf -l c:\Snort\log -i 2
        --== Initialization Complete ==--
   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.0.2-ODBC-MySQL-FlexRESP-WIN32 GRE (Build 92)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team
           Copyright (C) 1998-2010 Sourcefire, Inc., et al.
           Using PCRE version: 8.10 2010-06-25
           Using ZLIB version: 1.2.3
           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 1.12  <Build 18>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 4>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 3>
Commencing packet processing (pid=5896)
S5: Session exceeded configured max bytes to queue 1048576 using 1048979 bytes (
client queue). 192.168.168.7 11616 --> 92.46.53.163 80 (0) : LWstate 0x1 LWFlags
 0x2003
*** Caught Int-Signal
===============================================================================
Run time for packet processing was 5985.944000 seconds
Snort processed 11774 packets.
Snort ran for 0 days 1 hours 39 minutes 45 seconds
    Pkts/hr:        11774
   Pkts/min:          118
   Pkts/sec:            1
S5: Pruned session from cache that was using 1098947 bytes (purge whole cache).
192.168.168.7 11616 --> 92.46.53.163 80 (0) : LWstate 0x1 LWFlags 0x222003
===============================================================================
Packet I/O Totals:
   Received:       147490
   Analyzed:        11774 (  7.983%)
    Dropped:       135707 ( 92.011%)
   Filtered:            0 (  0.000%)
Outstanding:       135716 ( 92.017%)
   Injected:            0
===============================================================================
```

FIGURE 17.14: Working of Snort in Command Promt



How Snort Works

Reporting and Alerting Engine (ACID)

Databases

Web Servers

Primary NIC

Administrator

Snort Engine

NIC in Promicuous mode sniffing network traffic

Decoder

Base Detection Engine

Dynamic Loaded Libraries

Output Plugins

Rule Set

- **Decoder:** Saves the captured packets into a heap, identifies link level protocols, and decodes IP
- **Detection Engine:** It matches packets against rules previously saved in memory
- **Output Plug-ins:** These modules format notifications so operators can access in a variety of ways (console, extern files, databases, etc.)

Rules Files: These are plain text files which contain a list of rules with a known syntax

How Snort Works

The following are the three essential elements of the Snort tool:

- **Decoder:** Saves the **captured packets** into heap, identifies link level protocols, and decodes IP.
- **Detection Engine**: **Matches packets** against rules previously charged into memory since Snort initialization.
- **Output Plug-ins:** These modules format the notifications for the user to access them in different ways (console, extern files, databases, etc.).

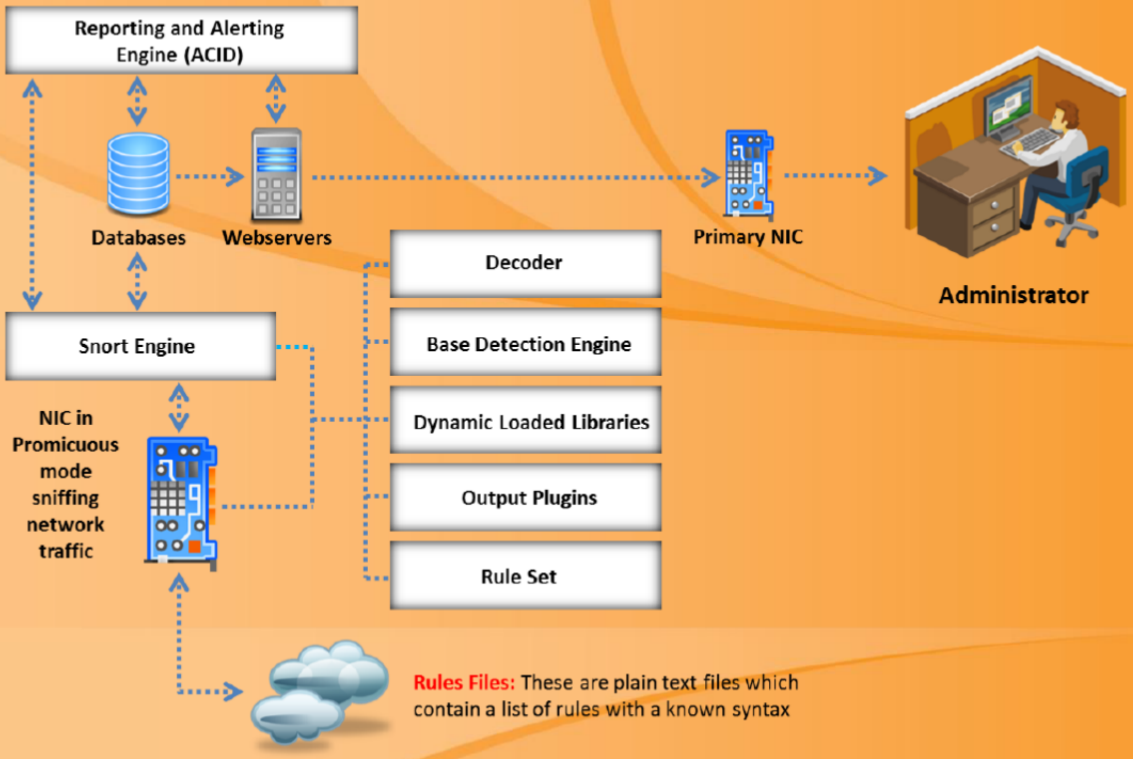
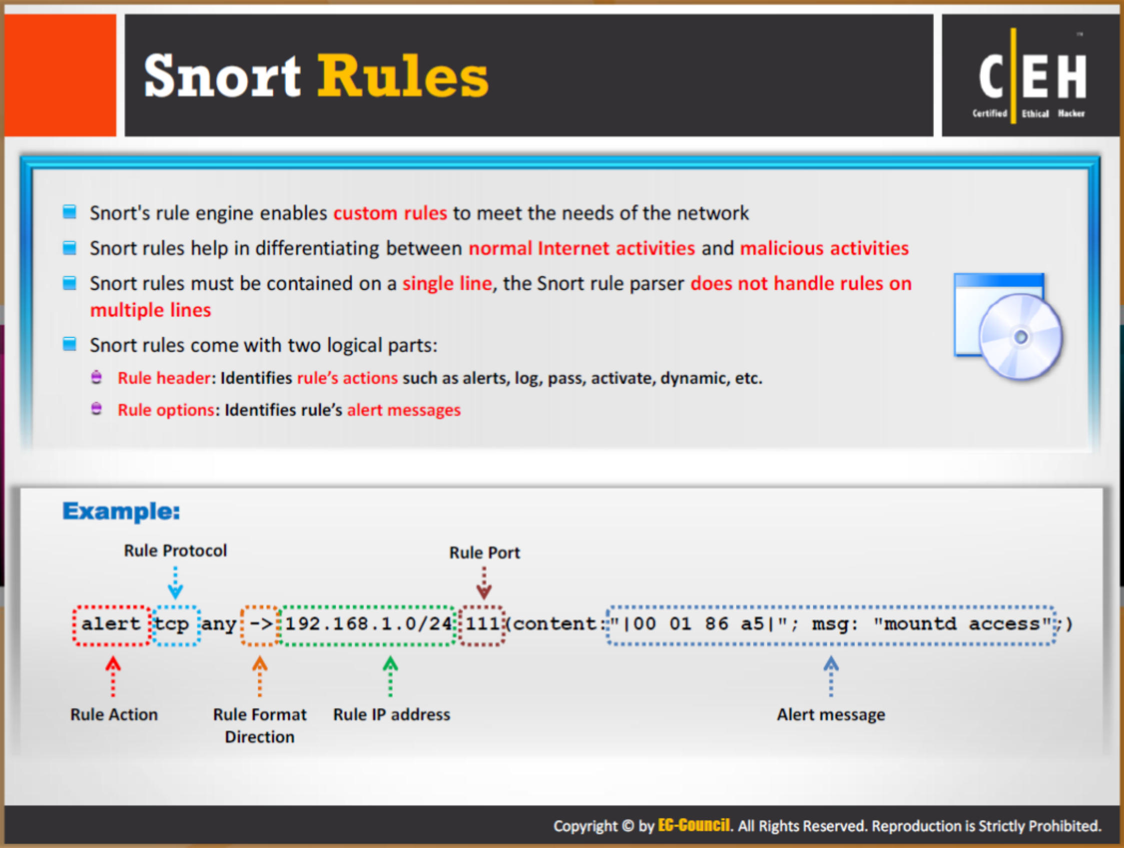

Reporting and Alerting Engine (ACID)
Databases
Webservers
Primary NIC
Administrator
Snort Engine
Decoder
Base Detection Engine
Dynamic Loaded Libraries
Output Plugins
Rule Set
NIC in Promicuous mode sniffing network traffic
Rules Files: These are plain text files which contain a list of rules with a known syntax

FIGURE 17.15: How Snort Works



Snort Rules

- Snort's rule engine enables custom rules to meet the needs of the network
- Snort rules help in differentiating between normal Internet activities and malicious activities
- Snort rules must be contained on a single line, the Snort rule parser does not handle rules on multiple lines
- Snort rules come with two logical parts:
  - Rule header: Identifies rule's actions such as alerts, log, pass, activate, dynamic, etc.
  - Rule options: Identifies rule's alert messages

Example:

```
alert tcp any -> 192.168.1.0/24 111 (content:"|00 01 86 a5|"; msg: "mountd access";)
```

Snort Rules

Snort uses the popular **libpcap library** (for UNIX/Linux) or **Winpcap** (for Windows), the same library that tcpdump uses to perform its packet sniffing. Snort decodes all the packets passing through the **network media** to which it is attached by entering promiscuous mode. Based on the content of the individual packets and rules defined in the configuration file, an alert is generated.

There are a number of rules that Snort allows the user to write. In addition, each of these Snort rules must describe the following:

- Any violation of the **security policy** of the company that might be a threat to the security of the company's network and other valuable information
- All the well-known and common attempts to exploit the **vulnerabilities** in the company's network
- The conditions in which a user thinks that a network packet(s) is unusual, i.e., if the identity of the packet is not authentic

Snort rules, written for both protocol analysis and content searching and matching, should be robust and flexible. The rules should be "**robust**"; it means the system should keep a rigid check on the activities taking place on the network and notify the administrator of any potential intrusion attempt. The rules should be "**flexible**"; it means that the system must be compatible enough to act immediately and take necessary remedial measures, according to the nature of the intrusion.

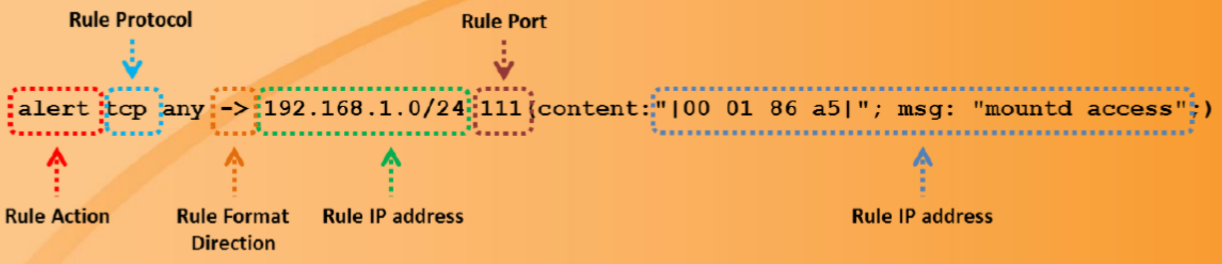
Both flexibility and robustness can be achieved using an easy-to-understand and lightweight **rule-description language** that aids in writing simple Snort rules. There are two basic principles that must be kept in mind while writing Snort rules. They are as follows:

- No written rule must extend beyond a single line, so rules should be short, precise, and easy-to-understand.
- Each rule should be divided into two logical sections:
  - The rule header
  - The rule options

The rule header contains the rule's action, the protocol, the source and destination IP addresses the source and destination port information, and the **CIDR (Classless Inter-Domain Routing) block**.

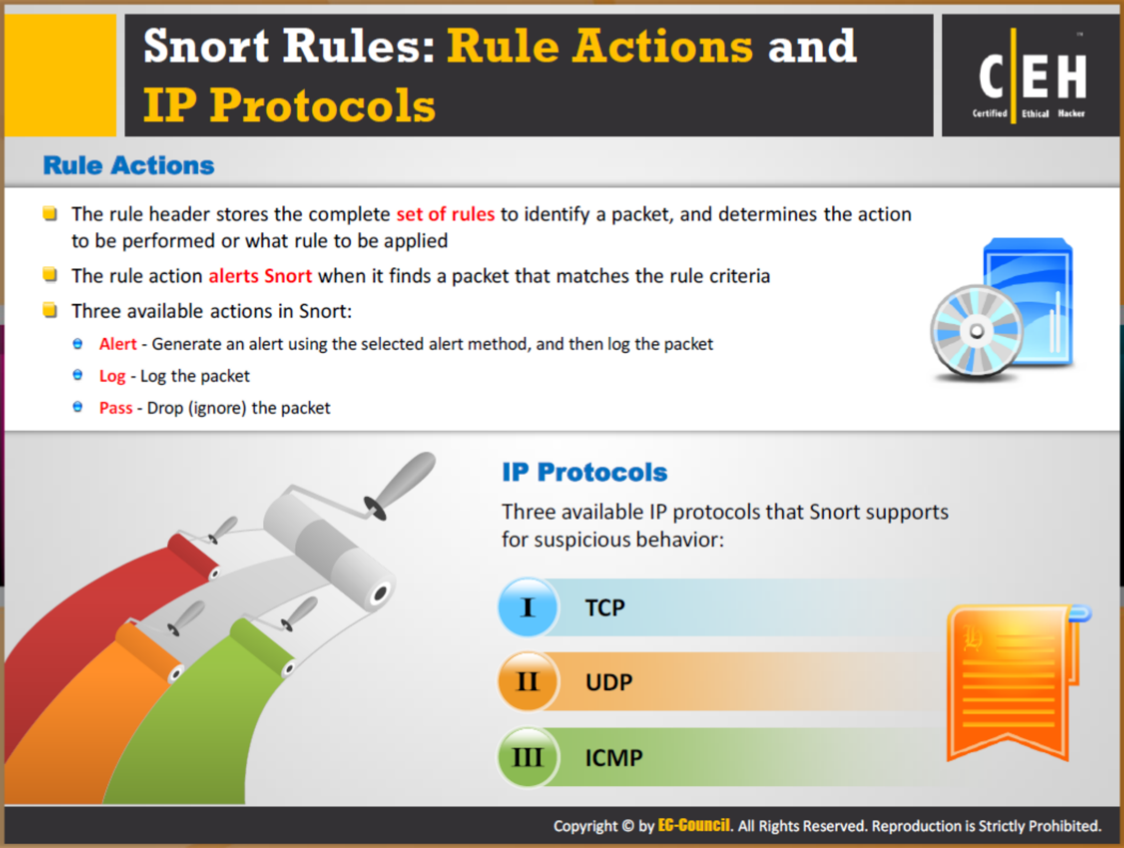
The rule option section includes alert messages, in addition to information about which part of the packet should be inspected in order to determine whether the rule action should be taken.

The following illustrates a sample example of a Snort rule:



```
alert tcp any -> 192.168.1.0/24 111 (content:"|00 01 86 a5|"; msg: "mountd access";)
```

FIGURE 17.16: Rules for Snort



Snort Rules: Rule Actions and IP Protocols

CEH
Certified Ethical Hacker

Rule Actions

- The rule header stores the complete set of rules to identify a packet, and determines the action to be performed or what rule to be applied
- The rule action alerts Snort when it finds a packet that matches the rule criteria
- Three available actions in Snort:
  - Alert - Generate an alert using the selected alert method, and then log the packet
  - Log - Log the packet
  - Pass - Drop (ignore) the packet

IP Protocols

Three available IP protocols that Snort supports for suspicious behavior:

I TCP
II UDP
III ICMP

Snort Rules: Rule Actions and IP Protocols

Source: http://manual.snort.org

The rule header contains the **information** that defines the who, where, and what of a packet, as well as what to do in the event that a packet with all the attributes indicated in the rule should show up. The first item in a rule is the rule action. The rule action tells Snort "**what to do**" when it finds a packet that matches the rule criteria. There are five available default actions in Snort: alert, log, pass, activate, and dynamic. In addition, if you are running Snort in inline mode, you have additional options which include drop, reject, and drop.

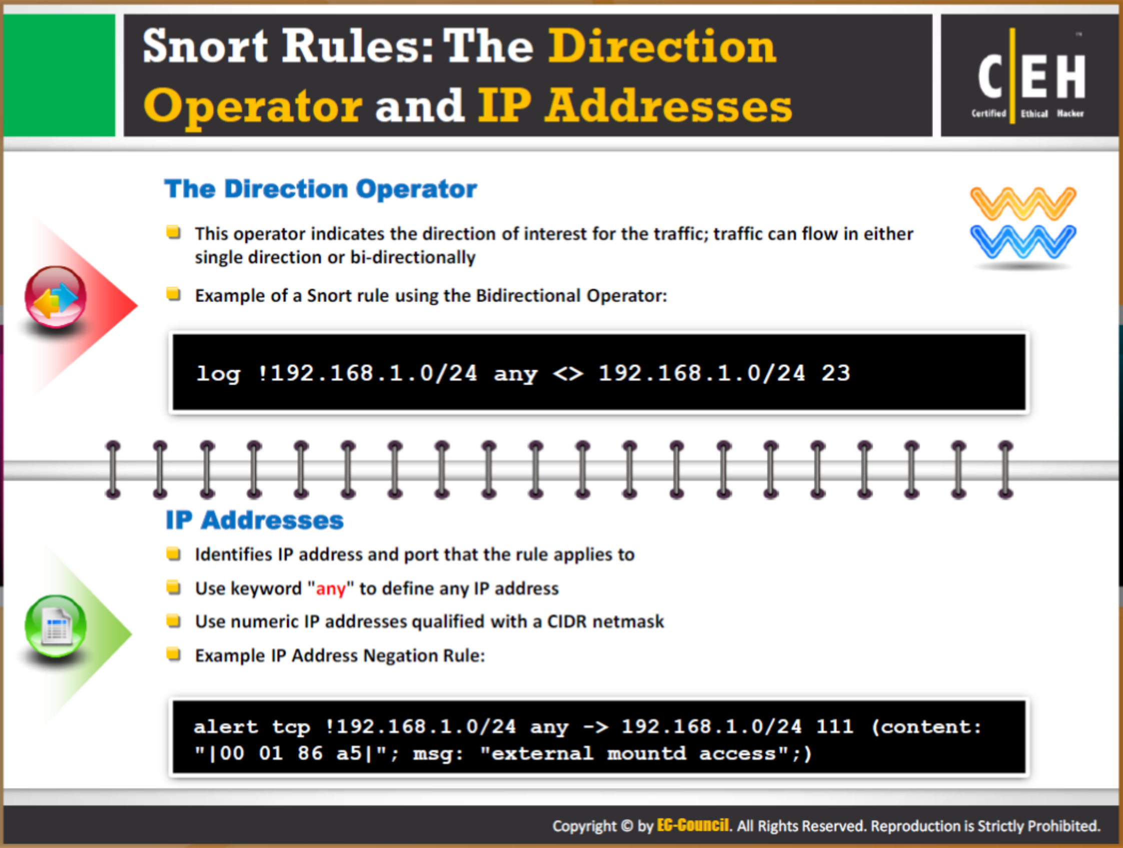
- **Alert** - generate an alert using the selected alert method, and then log the packet
- **Log** - log the packet
- **Pass** - ignore the packet
- **Activate** - alert and then turn on another dynamic rule
- **Dynamic** - remain idle until activated by an activate rule, then act as a log rule
- **Drop** - block and log the packet
- **Reject** - block the packet, log it, and then send a **TCP reset** if the protocol is TCP or an ICMP port unreachable message if the protocol is UDP

- **Sdrop** - block the packet but do not log it

The **Internet protocol (IP)** is used to send data from one system to another via the Internet. The IP supports unique addressing for every computer on a network. Data on the Internet protocol network is organized into packets. Each packet contains message data, source, destination, etc.

Three available IP protocols that Snort supports for suspicious behavior:

- **TCP:** TCP (transmission control protocol) is a part of the Internet Protocol. TCP is used to connect two different hosts and exchanges data between them.
- **UDP:** UDP, the acronym of User Datagram Protocol, is for broadcasting messages over a network.
- **ICMP:** The Internet Control Message protocol (ICMP) is a part of the Internet protocol. It is used by the operating systems in a network to send error messages, etc.



Snort Rules: The Direction Operator and IP Addresses

The Direction Operator

- This operator indicates the direction of interest for the traffic; traffic can flow in either single direction or bi-directionally
- Example of a Snort rule using the Bidirectional Operator:

```
log !192.168.1.0/24 any <> 192.168.1.0/24 23
```

IP Addresses

- Identifies IP address and port that the rule applies to
- Use keyword "any" to define any IP address
- Use numeric IP addresses qualified with a CIDR netmask
- Example IP Address Negation Rule:

```
alert tcp !192.168.1.0/24 any -> 192.168.1.0/24 111 (content:
"|00 01 86 a5|"; msg: "external mountd access";)
```

Snort Rules: The Direction Operator and IP Addresses

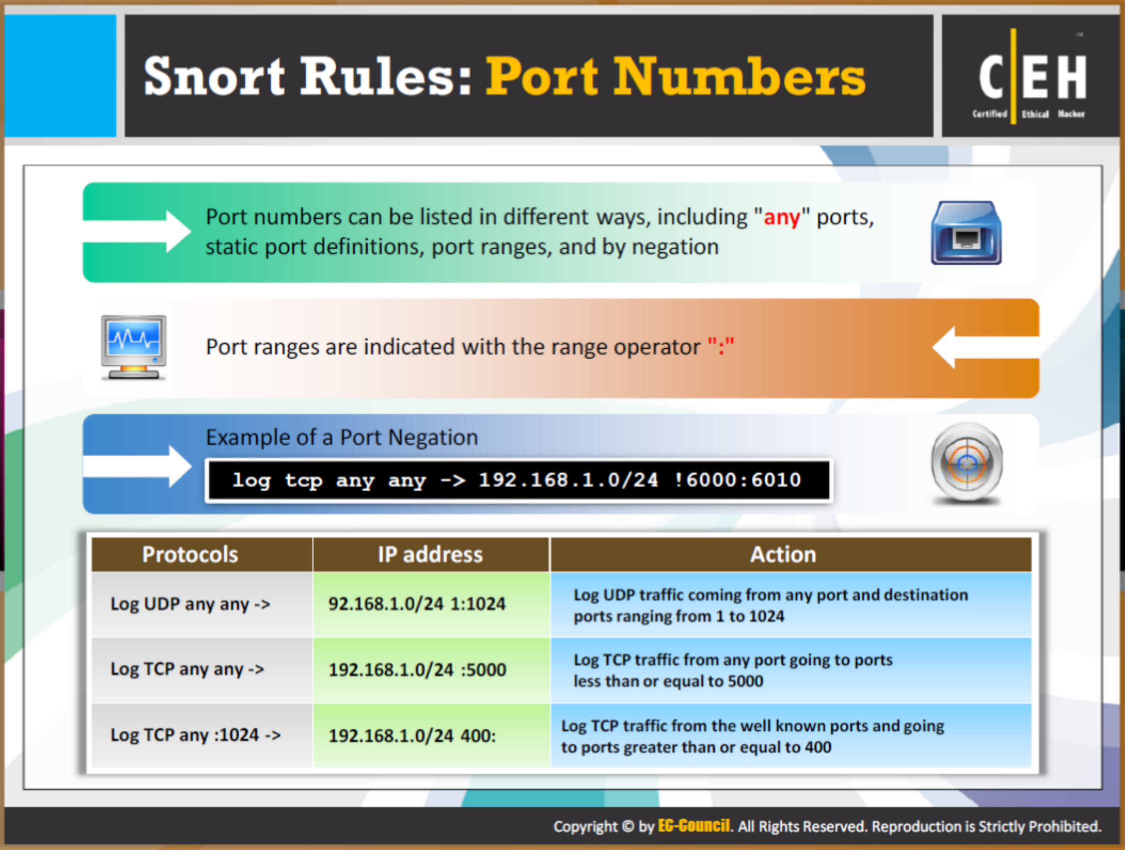
The direction operator -$>$ indicates the orientation, or direction, of the traffic that the rule applies to. The IP address and port numbers on the left side of the direction operator is considered to be the traffic coming from the source host, and the address and port information on the right side of the operator is the destination host. There is also a bidirectional operator, which is indicated with a $<>$ symbol. This tells Snort to consider the address/port pairs in either the source or destination orientation. This is handy for recording/analyzing both sides of a conversation, such as telnet or POP3 sessions.

Also, note that there is no $<$- operator. In Snort versions before 1.8.7, the direction operator did not have proper error checking and many people used an invalid token. The reason the $<$- does not exist is so that rules always read consistently.

The next fields in a Snort rule are used to specify the source and destination IP addresses and ports of the packet, as well as the direction in which the packet is traveling. Snort can accept a single IP address or a list of addresses. When specifying a list of IP address, you should separate each one with a comma and then enclose the list within square brackets, like this:

[192.168.1.1,192.168.1.45,10.1.1.24]

When doing this, be careful not to use any whitespace. You can also specify ranges of IP addresses using **CIDR notation**, or even include CIDR ranges within lists. Snort also allows you to apply the logical NOT operator (!) to an IP address or **CIDR range** to specify that the rule should match all but that address or range of addresses.



Snort Rules: Port Numbers

Port numbers can be listed in different ways, including "any" ports, static port definitions, port ranges, and by negation

Port ranges are indicated with the range operator ":"

Example of a Port Negation

```
log tcp any any -> 192.168.1.0/24 !6000:6010
```

Protocols	IP address	Action
Log UDP any any ->	92.168.1.0/24 1:1024	Log UDP traffic coming from any port and destination ports ranging from 1 to 1024
Log TCP any any ->	192.168.1.0/24 :5000	Log TCP traffic from any port going to ports less than or equal to 5000
Log TCP any :1024 ->	192.168.1.0/24 400:	Log TCP traffic from the well known ports and going to ports greater than or equal to 400

Snort Rules: Port Numbers

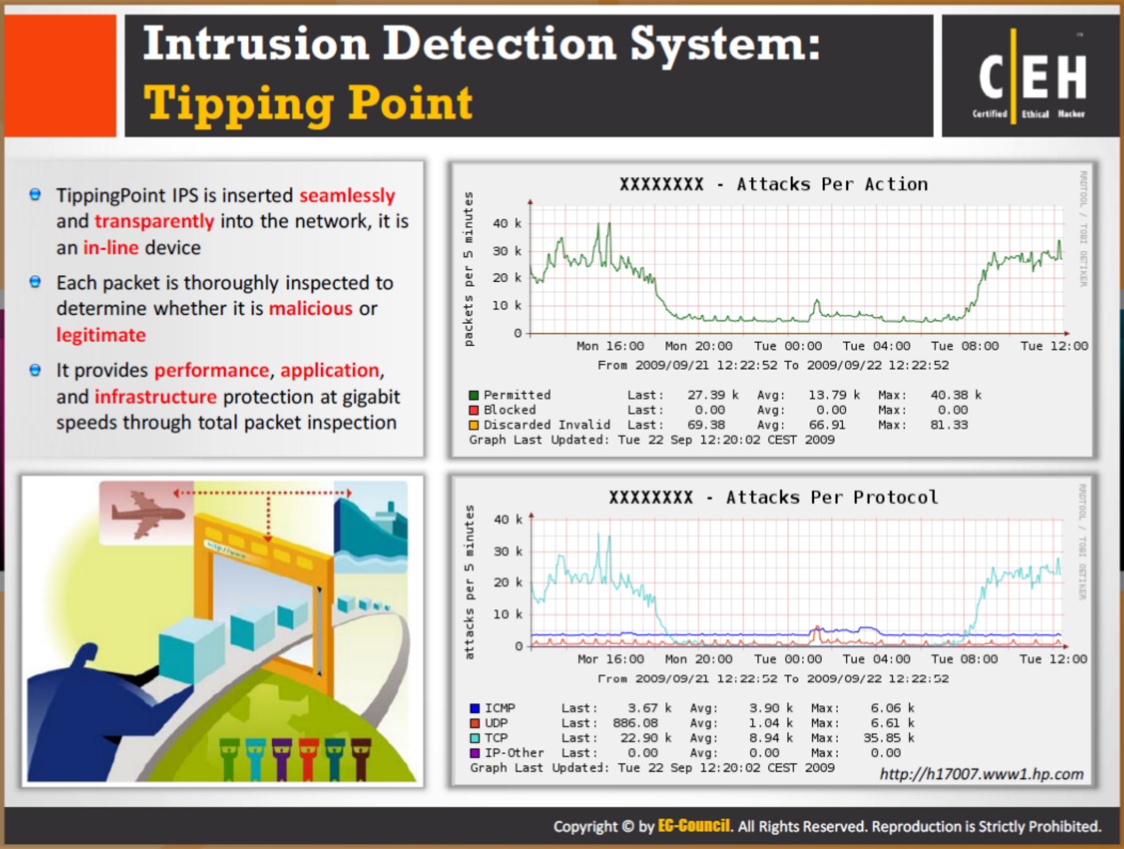
Port numbers may be specified in a number of ways, including any ports, static port definitions, ranges, and by negation. Any ports are a wildcard value, meaning literally any port. Static ports are indicated by a single port number, such as **111 for portmapper**, **23 for telnet**, or **80 for http**, etc. Port ranges are indicated with the range operator ":" The range operator may be applied in a number of ways to take on different meanings.

Example of Port Negation:

```
log tcp any any -> 192.168.1.0/24 !6000:6010
```

Protocols	IP address	Action
Log UDP any any ->	92.168.1.0/24 1:1024	Log UDP traffic coming from any port and destination ports ranging from 1 to 1024
Log TCP any any ->	192.168.1.0/24 :5000	Log TCP traffic from any port going to ports less than or equal to 5000
Log TCP any :1024 ->	192.168.1.0/24 400:	Log TCP traffic from privileged ports less than or equal to 1024 going to ports greater than or equal to 400

TABLE 17.1: Port Numbers



Intrusion Detection System: Tipping Point

- TippingPoint IPS is inserted seamlessly and transparently into the network, it is an in-line device
- Each packet is thoroughly inspected to determine whether it is malicious or legitimate
- It provides performance, application, and infrastructure protection at gigabit speeds through total packet inspection

http://www
XXXXXXXX - Attacks Per Action
packets per 5 minutes
40 k
30 k
20 k
10 k
0
Mon 16:00
Mon 20:00
Tue 00:00
Tue 04:00
Tue 08:00
Tue 12:00
RRDTOOL / TOBI OETIKER

```
From 2009/09/21 12:22:52 To 2009/09/22 12:22:52

■ Permitted          Last:   27.39 k   Avg:   13.79 k   Max:   40.38 k
■ Blocked            Last:    0.00     Avg:    0.00     Max:    0.00
■ Discarded Invalid  Last:   69.38     Avg:   66.91     Max:   81.33
Graph Last Updated: Tue 22 Sep 12:20:02 CEST 2009
```

XXXXXXXX - Attacks Per Protocol
attacks per 5 minutes
40 k
30 k
20 k
10 k
0
Mor 16:00
Mon 20:00
Tue 00:00
Tue 04:00
Tue 08:00
Tue 12:00
RRDTOOL / TOBI OETIKER

```
From 2009/09/21 12:22:52 To 2009/09/22 12:22:52

■ ICMP      Last:   3.67 k   Avg:   3.90 k   Max:    6.06 k
■ UDP       Last: 886.08     Avg:   1.04 k   Max:    6.61 k
■ TCP       Last:  22.90 k   Avg:   8.94 k   Max:   35.85 k
■ IP-Other  Last:   0.00     Avg:   0.00     Max:    0.00
Graph Last Updated: Tue 22 Sep 12:20:02 CEST 2009
```

http://h17007.www1.hp.com

Intrusion Detection System: Tipping Point

Source: http://h10163.www1.hp.com

TippingPoint IPS is inserted **seamlessly** and **transparently** into the network; it is an in-line device. Each packet is thoroughly inspected to determine whether it is malicious or legitimate. It provides performance, application, and infrastructure protection at gigabit speeds through total packet inspection.

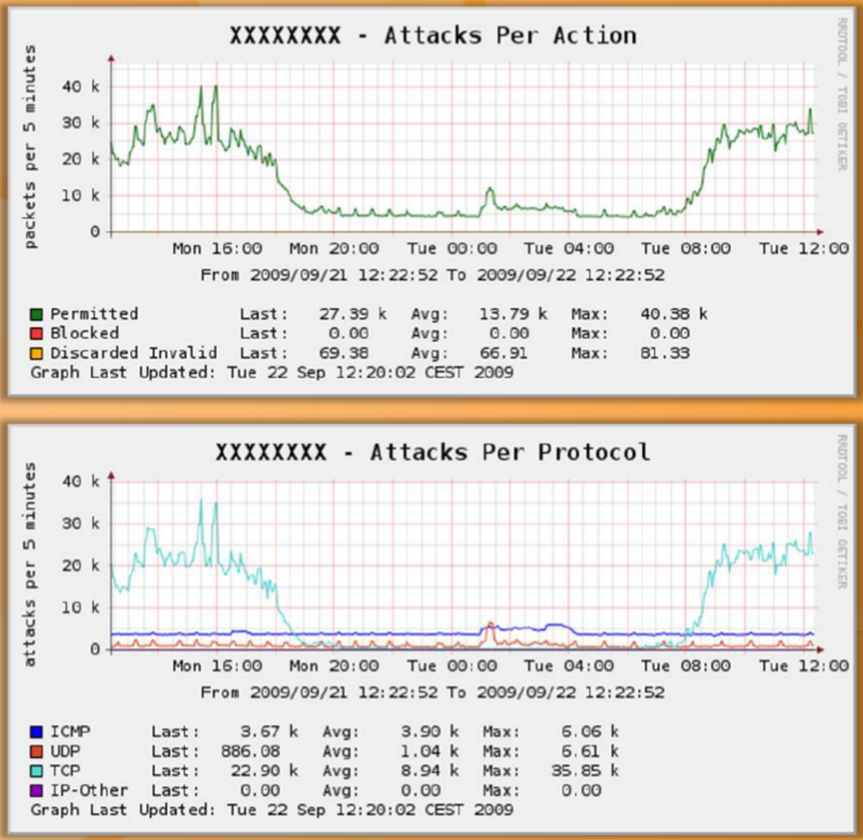
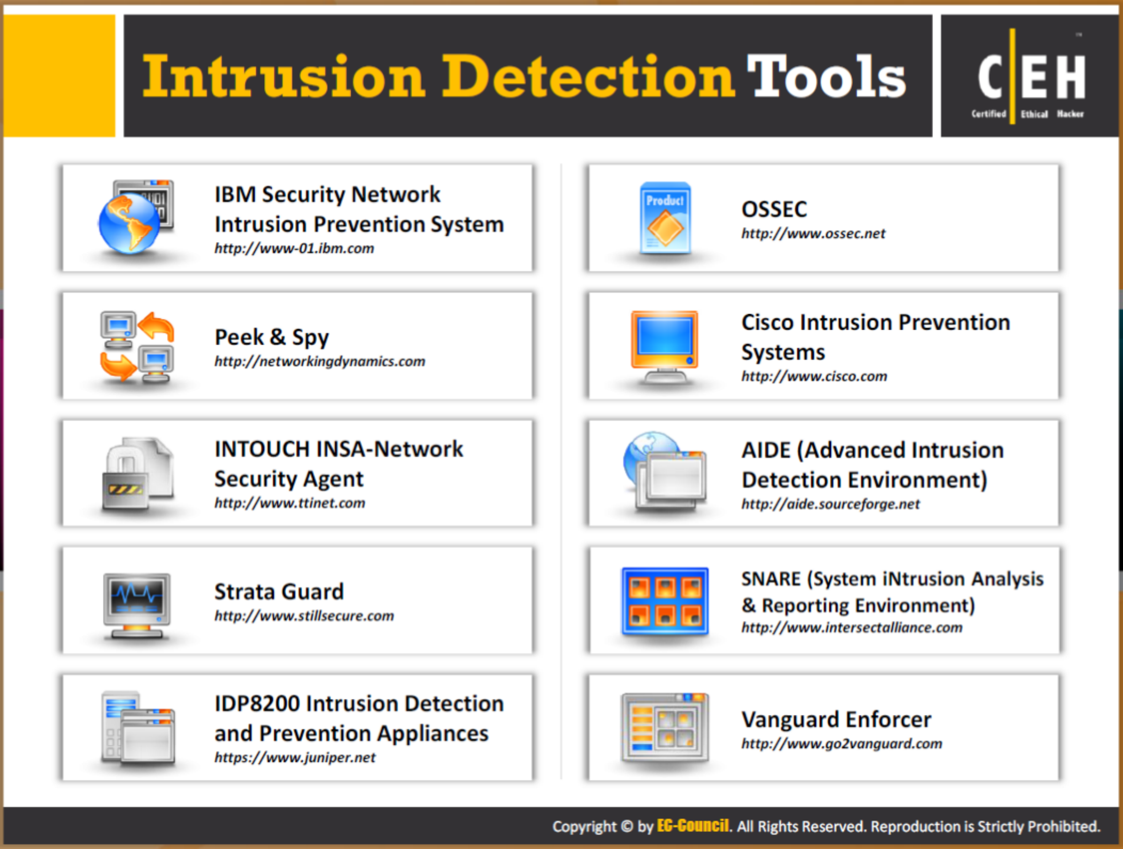


FIGURE 17.17: Tipping Point Screenshot



Intrusion Detection Tools

C|EH
Certified Ethical Hacker

IBM Security Network Intrusion Prevention System
http://www-01.ibm.com

Peek & Spy
http://networkingdynamics.com

INTOUCH INSA-Network Security Agent
http://www.ttinet.com

Strata Guard
http://www.stillsecure.com

IDP8200 Intrusion Detection and Prevention Appliances
https://www.juniper.net

OSSEC
http://www.ossec.net

Cisco Intrusion Prevention Systems
http://www.cisco.com

AIDE (Advanced Intrusion Detection Environment)
http://aide.sourceforge.net

SNARE (System iNtrusion Analysis & Reporting Environment)
http://www.intersectalliance.com

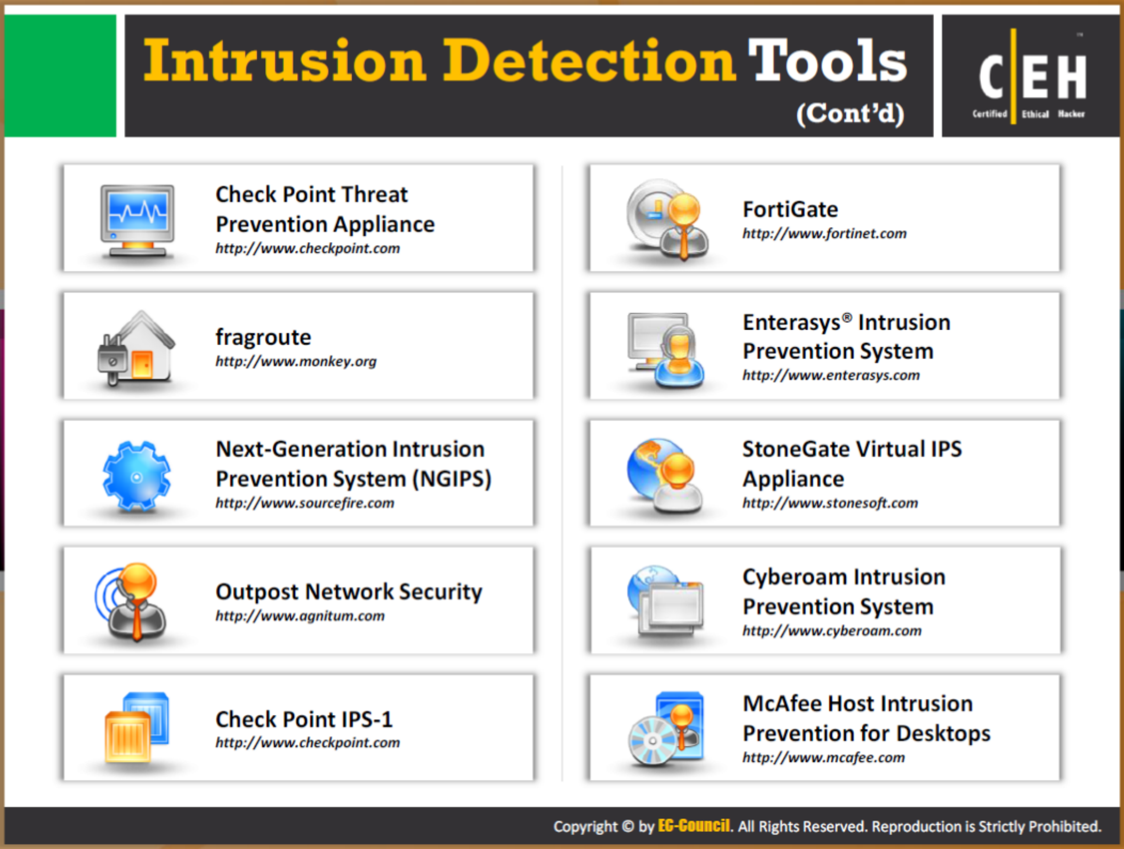
Vanguard Enforcer
http://www.go2vanguard.com

Intrusion Detection Tools

Intrusion detection tools detect anomalies. These tools, when run on a dedicated workstation, read all network packets, reconstruct user sessions, and scan for possible intrusions by looking for attack signatures and **network traffic statistical anomalies**. In addition, these tools give real-time, zero-day protection from network attacks and malicious traffic, and prevent malware, spyware, port scans, viruses, and **DoS** and **DDoS** from compromising hosts. A few of intrusion detection tools are listed as follows:

- IBM Security Network Intrusion Prevention System available at http://www-01.ibm.com
- Peek & Spy available at http://networkingdynamics.com
- INTOUCH INSA-Network Security Agent available at http://www.ttinet.com
- Strata Guard available at http://www.stillsecure.com
- IDP8200 Intrusion Detection and Prevention Appliances available at https://www.juniper.net
- OSSEC available at http://www.ossec.net
- Cisco Intrusion Prevention Systems available at http://www.cisco.com

- AIDE (Advanced Intrusion Detection Environment) available at http://aide.sourceforge.net
- SNARE (System iNtrusion Analysis & Reporting Environment) available at
- http://www.intersectalliance.com
- Vanguard Enforcer available at http://www.go2vanguard.com



Intrusion Detection Tools (Cont'd)

Check Point Threat Prevention Appliance
http://www.checkpoint.com

fragroute
http://www.monkey.org

Next-Generation Intrusion Prevention System (NGIPS)
http://www.sourcefire.com

Outpost Network Security
http://www.agnitum.com

Check Point IPS-1
http://www.checkpoint.com

FortiGate
http://www.fortinet.com

Enterasys® Intrusion Prevention System
http://www.enterasys.com

StoneGate Virtual IPS Appliance
http://www.stonesoft.com

Cyberoam Intrusion Prevention System
http://www.cyberoam.com

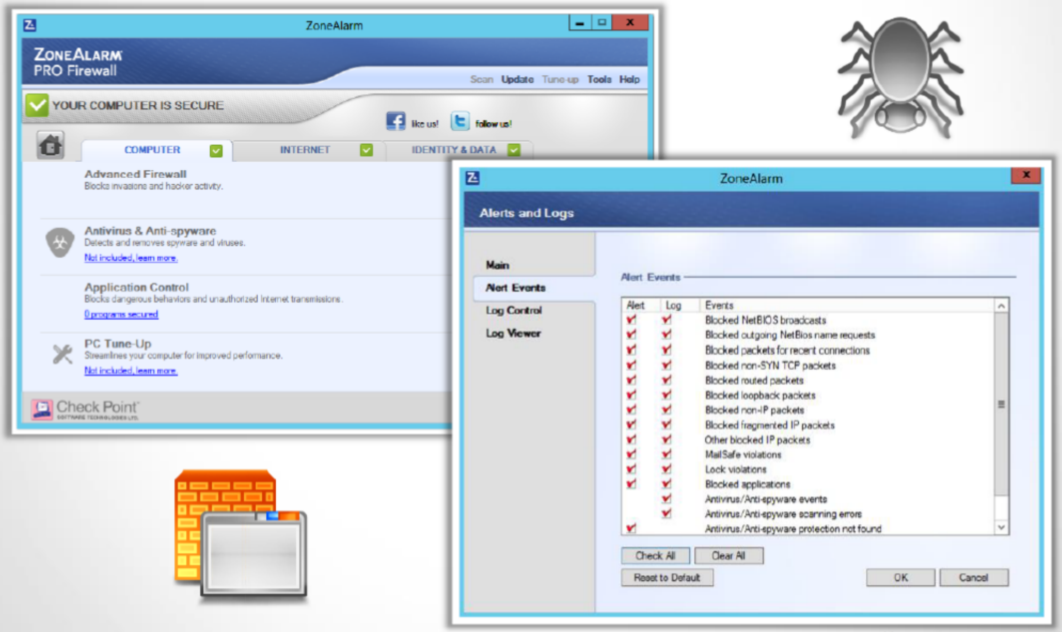
McAfee Host Intrusion Prevention for Desktops
http://www.mcafee.com

Intrusion Detection Tools (Cont'd)

In addition, to the previously mentioned intrusion detection tools, there are few more tools that can be used for detecting intrusions:

- Check Point Threat Prevention Appliance available at http://www.checkpoint.com
- Fragroute available at http://www.monkey.org
- Next-Generation Intrusion Prevention System (NGIPS) available at http://www.sourcefire.com
- Outpost Network Security available at http://www.agnitum.com
- Check Point IPS-1 available at http://www.checkpoint.com
- FortiGate available at http://www.fortinet.com
- Enterasys® Intrusion Prevention System available at http://www.enterasys.com
- StoneGate Virtual IPS Appliance available at http://www.stonesoft.com
- Cyberoam Intrusion Prevention System available at http://www.cyberoam.com
- McAfee Host Intrusion Prevention for Desktops available at http://www.mcafee.com

Firewall: ZoneAlarm PRO Firewall


ZoneAlarm
ZONEALARM
PRO Firewall
Scan
Update
Tune-up
Tools
Help
YOUR COMPUTER IS SECURE
like us!
follow us!
COMPUTER
INTERNET
IDENTITY & DATA
Check Point
SOFTWARE TECHNOLOGIES LTD.

Advanced Firewall
Blocks invasions and hacker activity.

Antivirus & Anti-spyware
Detects and removes spyware and viruses.
Not included, learn more.

Application Control
Blocks dangerous behaviors and unauthorized Internet transmissions.
0 programs secured

PC Tune-Up
Streamlines your computer for improved performance.
Not included, learn more.

ZoneAlarm — Alerts and Logs

Main
Alert Events
Log Control
Log Viewer

Alert Events

Alert	Log	Events
✓	✓	Blocked NetBIOS broadcasts
✓	✓	Blocked outgoing NetBios name requests
✓	✓	Blocked packets for recent connections
✓	✓	Blocked non-SYN TCP packets
✓	✓	Blocked routed packets
✓	✓	Blocked loopback packets
✓	✓	Blocked non-IP packets
✓	✓	Blocked fragmented IP packets
✓	✓	Other blocked IP packets
✓	✓	MailSafe violations
✓	✓	Lock violations
✓	✓	Blocked applications
	✓	Antivirus/Anti-spyware events
	✓	Antivirus/Anti-spyware scanning errors
✓		Antivirus/Anti-spyware protection not found

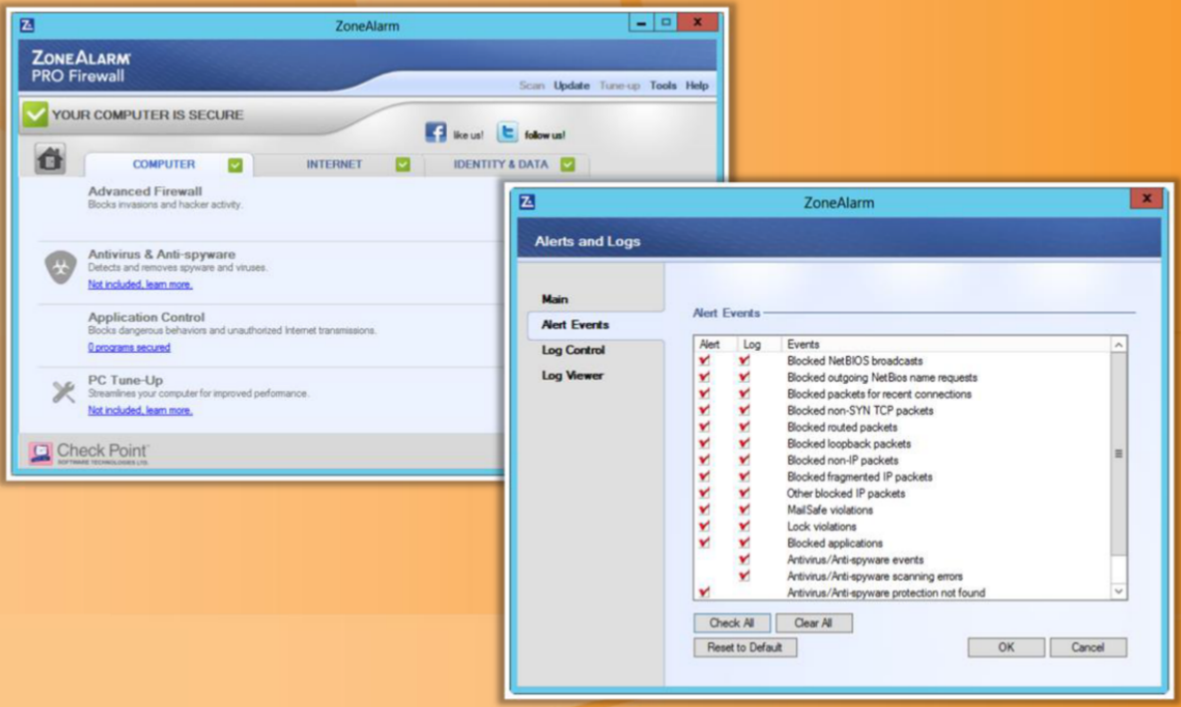
Check All
Clear All
Reset to Default
OK
Cancel

http://www.zonealarm.com

Firewall: ZoneAlarm PRO Firewall

Source: http://www.zonealarm.com

ZoneAlarm PRO Firewall **blocks attackers** and intruders from accessing your system. It monitors programs for suspicious behavior, spotting and stopping new attacks that bypass traditional antivirus protection. It prevents identity theft by guarding your personal data. It even erases your tracks allowing you to surf the web in complete privacy. Furthermore, it locks out attackers, blocks intrusions, and makes your **PC invisible online**. In addition, it filters out annoying and potentially dangerous email.


ZoneAlarm
ZONEALARM PRO Firewall
Scan
Update
Tune-up
Tools
Help
YOUR COMPUTER IS SECURE
like us!
follow us!
COMPUTER
INTERNET
IDENTITY & DATA
Check Point
ZoneAlarm
Alerts and Logs
Main
Alert Events
Log Control
Log Viewer
Alert Events
Check All
Clear All
Reset to Default
OK
Cancel

Advanced Firewall
Blocks invasions and hacker activity.

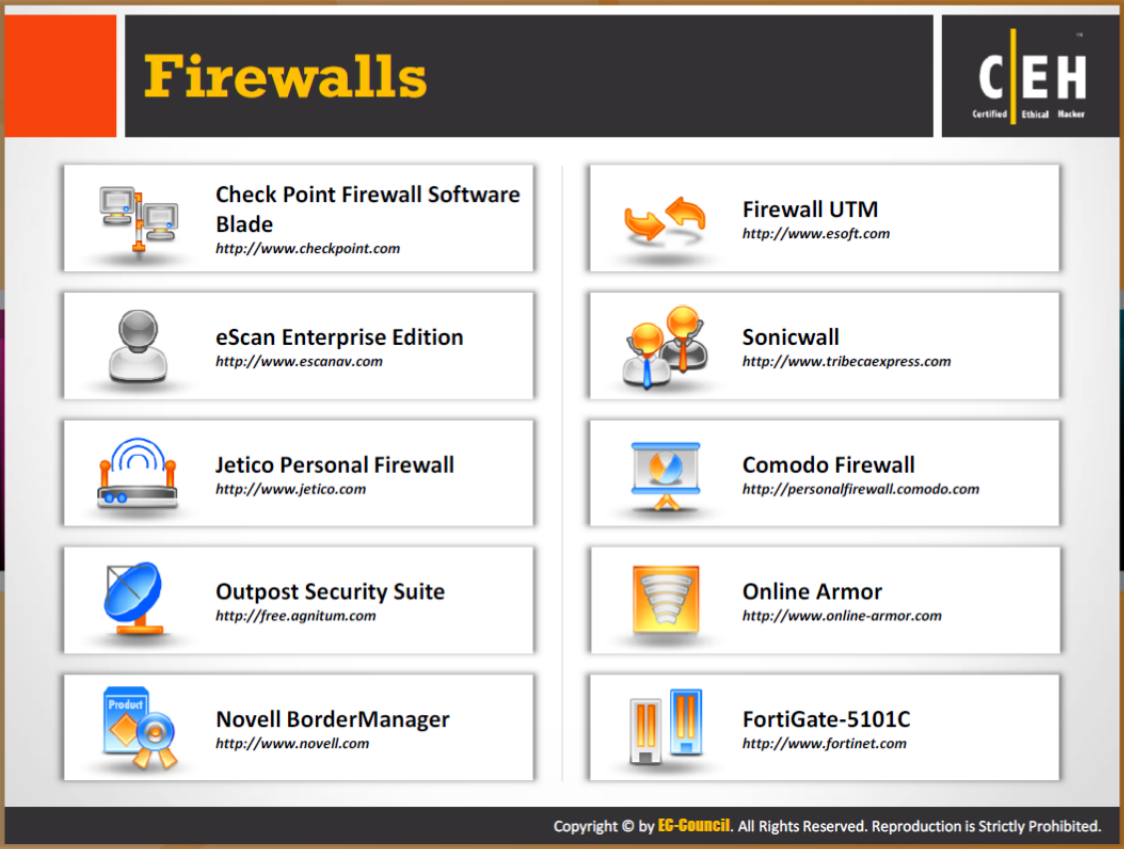
Antivirus & Anti-spyware
Detects and removes spyware and viruses.
Not included, learn more.

Application Control
Blocks dangerous behaviors and unauthorized Internet transmissions.
0 programs secured

PC Tune-Up
Streamlines your computer for improved performance.
Not included, learn more.

Alert	Log	Events
✓	✓	Blocked NetBIOS broadcasts
✓	✓	Blocked outgoing NetBios name requests
✓	✓	Blocked packets for recent connections
✓	✓	Blocked non-SYN TCP packets
✓	✓	Blocked routed packets
✓	✓	Blocked loopback packets
✓	✓	Blocked non-IP packets
✓	✓	Blocked fragmented IP packets
✓	✓	Other blocked IP packets
✓	✓	MailSafe violations
✓	✓	Lock violations
✓	✓	Blocked applications
	✓	Antivirus/Anti-spyware events
	✓	Antivirus/Anti-spyware scanning errors
✓		Antivirus/Anti-spyware protection not found

FIGURE 17.18: ZoneAlarm PRO Firewall Screenshot



Firewalls

Check Point Firewall Software Blade
http://www.checkpoint.com

eScan Enterprise Edition
http://www.escanav.com

Jetico Personal Firewall
http://www.jetico.com

Outpost Security Suite
http://free.agnitum.com

Novell BorderManager
http://www.novell.com

Firewall UTM
http://www.esoft.com

Sonicwall
http://www.tribecaexpress.com

Comodo Firewall
http://personalfirewall.comodo.com

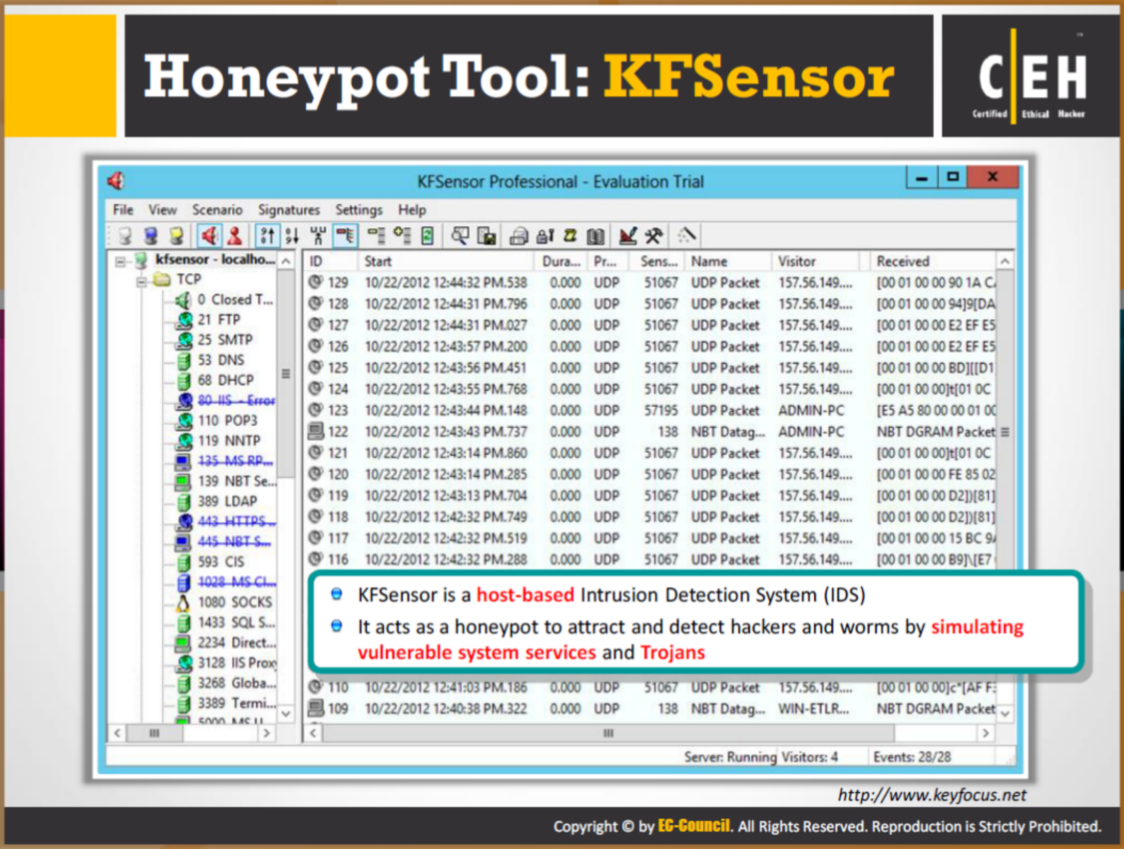
Online Armor
http://www.online-armor.com

FortiGate-5101C
http://www.fortinet.com

Firewalls

Firewalls provide essential protection to the **computers against viruses**, privacy threats, objectionable content, hackers, and malicious software when networked or connected to the Internet. A firewall **monitors running applications** that access the network. It analyzes downloads and warns you if downloading a malicious file, stops it from infecting your PC. A few of the firewalls that provide system protection are listed as follows:

- Check Point Firewall Software Blade available at http://www.checkpoint.com
- eScan Enterprise available at http://www.escanav.com
- Jetico Personal Firewall available at http://www.jetico.com
- Outpost Security Suite available at http://free.agnitum.com
- Novell BorderManager available at http://www.novell.com
- Firewall UTM available at http://www.esoft.com
- Sonicwall available at http://www.tribecaexpress.com
- Comodo Firewall available at http://personalfirewall.comodo.com
- Online Armor available at http://www.online-armor.com
- FortiGate-5101C available at http://www.fortinet.com



Honeypot Tool: KFSensor

KFSensor Professional - Evaluation Trial
File View Scenario Signatures Settings Help
kfsensor - localho...
TCP
0 Closed T...
21 FTP
25 SMTP
53 DNS
68 DHCP
~~80 IIS - Error~~
110 POP3
119 NNTP
~~135 MS RP...~~
139 NBT Se...
389 LDAP
~~443 HTTPS...~~
~~445 NBT S...~~
593 CIS
~~1028 MS Cl...~~
1080 SOCKS
1433 SQL S...
2234 Direct...
3128 IIS Prox
3268 Globa...
3389 Termi...

ID	Start	Dura...	Pr...	Sens...	Name	Visitor	Received
129	10/22/2012 12:44:32 PM.538	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 90 1A C
128	10/22/2012 12:44:31 PM.796	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 94]9[DA
127	10/22/2012 12:44:31 PM.027	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 E2 EF E5
126	10/22/2012 12:43:57 PM.200	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 E2 EF E5
125	10/22/2012 12:43:56 PM.451	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 BD][[D1
124	10/22/2012 12:43:55 PM.768	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00]t[01 0C
123	10/22/2012 12:43:44 PM.148	0.000	UDP	57195	UDP Packet	ADMIN-PC	[E5 A5 80 00 00 01 0
122	10/22/2012 12:43:43 PM.737	0.000	UDP	138	NBT Datag...	ADMIN-PC	NBT DGRAM Packet
121	10/22/2012 12:43:14 PM.860	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00]t[01 0C
120	10/22/2012 12:43:14 PM.285	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 FE 85 02
119	10/22/2012 12:43:13 PM.704	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 D2])[81]
118	10/22/2012 12:42:32 PM.749	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 D2])[81]
117	10/22/2012 12:42:32 PM.519	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 15 BC 9
116	10/22/2012 12:42:32 PM.288	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 B9]\[E7
110	10/22/2012 12:41:03 PM.186	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00]c*[AF F
109	10/22/2012 12:40:38 PM.322	0.000	UDP	138	NBT Datag...	WIN-ETLR...	NBT DGRAM Packet

Server: Running Visitors: 4 Events: 28/28

- KFSensor is a host-based Intrusion Detection System (IDS)
- It acts as a honeypot to attract and detect hackers and worms by simulating vulnerable system services and Trojans

http://www.keyfocus.net

Honeypot Tool: KFSensor

Source: http://www.keyfocus.net

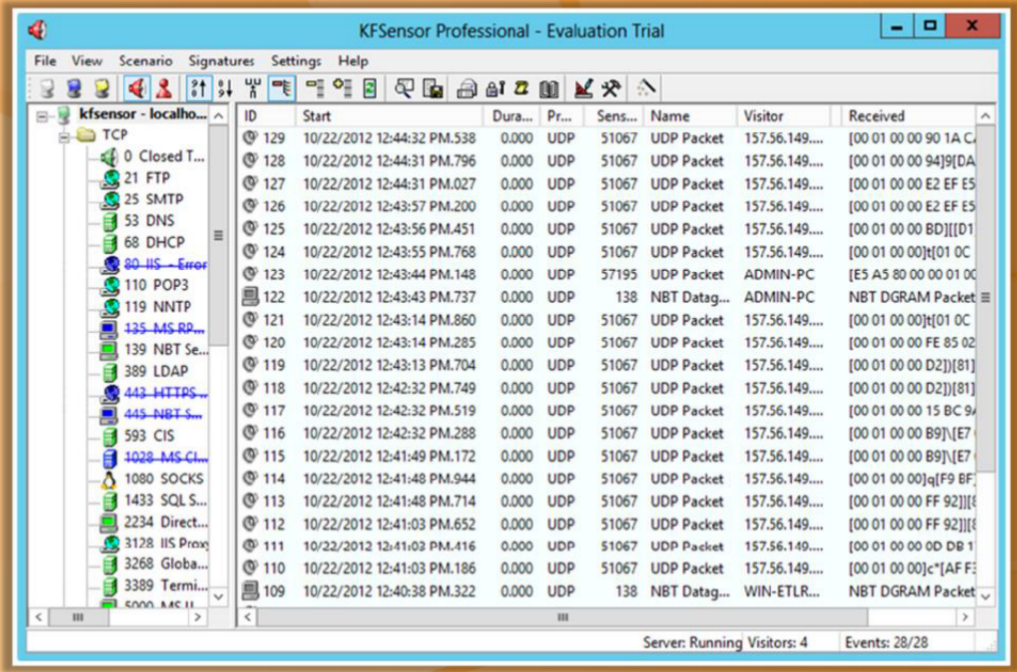
KFSensor is a **Windows-based honeypot intrusion detection system (IDS)**. It acts as a honeypot to attract and detect hackers and worms by simulating vulnerable system services and Trojans. By acting as a decoy server, it can divert attacks from critical systems and provide a higher level of information than can be achieved by using **firewalls** and **NIDS alone**.

KFSensor is designed for use in a Windows-based corporate environment and contains many innovative and unique features such as remote management, a Snort-compatible signature engine, and emulations of Windows networking protocols.

Features:

- GUI-based management console
- Remote management
- Snort compatible signature engine
- Emulations of Windows networking protocols
- Export logs in multiple formats

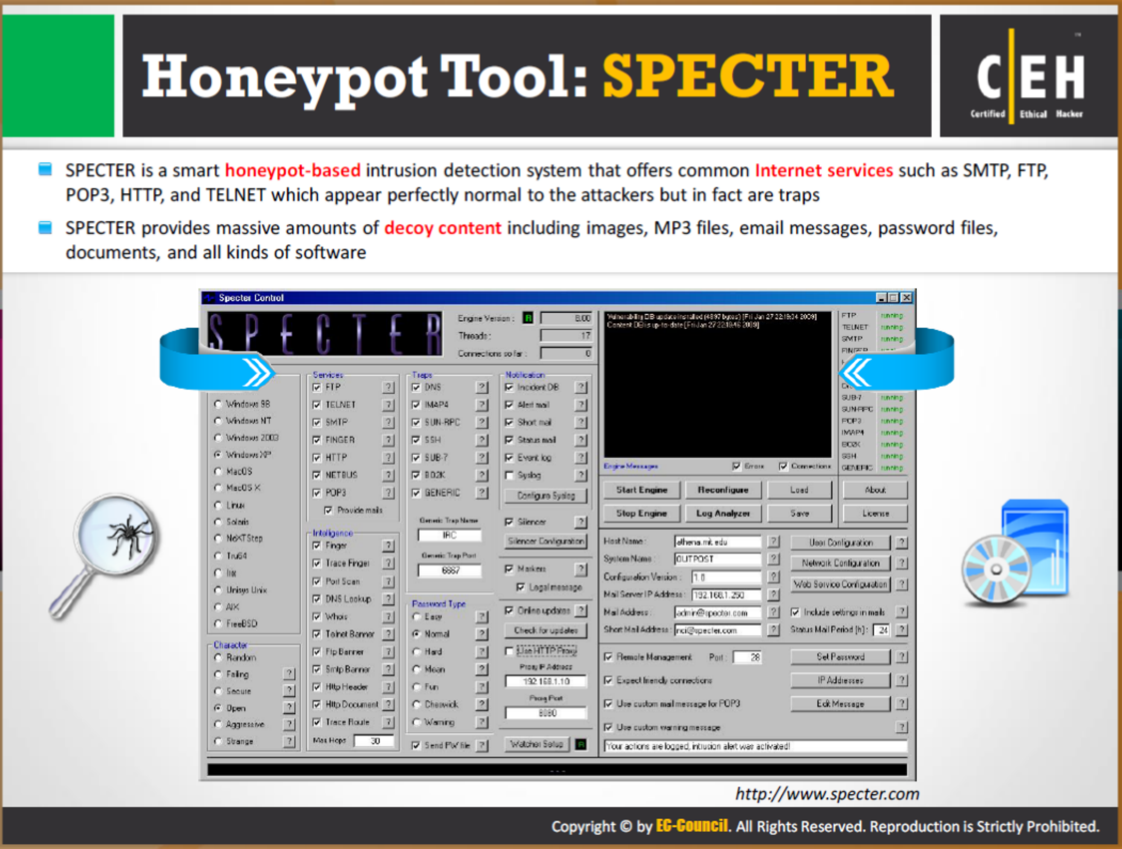
- Denial-of-service (DOS) attack protection


KFSensor Professional - Evaluation Trial
File
View
Scenario
Signatures
Settings
Help
kfsensor - localho...
TCP
0 Closed T...
21 FTP
25 SMTP
53 DNS
68 DHCP
~~80 IIS - Error~~
110 POP3
119 NNTP
~~135 MS RP...~~
139 NBT Se...
389 LDAP
~~443 HTTPS...~~
~~445 NBT S...~~
593 CIS
~~1028 MS Cl...~~
1080 SOCKS
1433 SQL S...
2234 Direct...
3128 IIS Prox
3268 Globa...
3389 Termi...
5000 MS U...

ID	Start	Dura...	Pr...	Sens...	Name	Visitor	Received
129	10/22/2012 12:44:32 PM.538	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 90 1A C
128	10/22/2012 12:44:31 PM.796	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 94]9[DA
127	10/22/2012 12:44:31 PM.027	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 E2 EF E5
126	10/22/2012 12:43:57 PM.200	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 E2 EF E5
125	10/22/2012 12:43:56 PM.451	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 BD][[D1
124	10/22/2012 12:43:55 PM.768	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00]t[01 0C
123	10/22/2012 12:43:44 PM.148	0.000	UDP	57195	UDP Packet	ADMIN-PC	[E5 A5 80 00 00 01 0
122	10/22/2012 12:43:43 PM.737	0.000	UDP	138	NBT Datag...	ADMIN-PC	NBT DGRAM Packet
121	10/22/2012 12:43:14 PM.860	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00]t[01 0C
120	10/22/2012 12:43:14 PM.285	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 FE 85 02
119	10/22/2012 12:43:13 PM.704	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 D2])[81]
118	10/22/2012 12:42:32 PM.749	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 D2])[81]
117	10/22/2012 12:42:32 PM.519	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 15 BC 9
116	10/22/2012 12:42:32 PM.288	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 B9]\[E7
115	10/22/2012 12:41:49 PM.172	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 B9]\[E7
114	10/22/2012 12:41:48 PM.944	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00]q[F9 BF
113	10/22/2012 12:41:48 PM.714	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 FF 92]][
112	10/22/2012 12:41:03 PM.652	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 FF 92]][
111	10/22/2012 12:41:03 PM.416	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00 0D DB 1
110	10/22/2012 12:41:03 PM.186	0.000	UDP	51067	UDP Packet	157.56.149....	[00 01 00 00]c*[AF F
109	10/22/2012 12:40:38 PM.322	0.000	UDP	138	NBT Datag...	WIN-ETLR...	NBT DGRAM Packet

Server: Running Visitors: 4 Events: 28/28

FIGURE 17.19: KFSensor Screenshot



Honeypot Tool: SPECTER

- SPECTER is a smart honeypot-based intrusion detection system that offers common Internet services such as SMTP, FTP, POP3, HTTP, and TELNET which appear perfectly normal to the attackers but in fact are traps
- SPECTER provides massive amounts of decoy content including images, MP3 files, email messages, password files, documents, and all kinds of software

http://www.specter.com

Honeypot Tool: SPECTER

Source http://www.specter.com

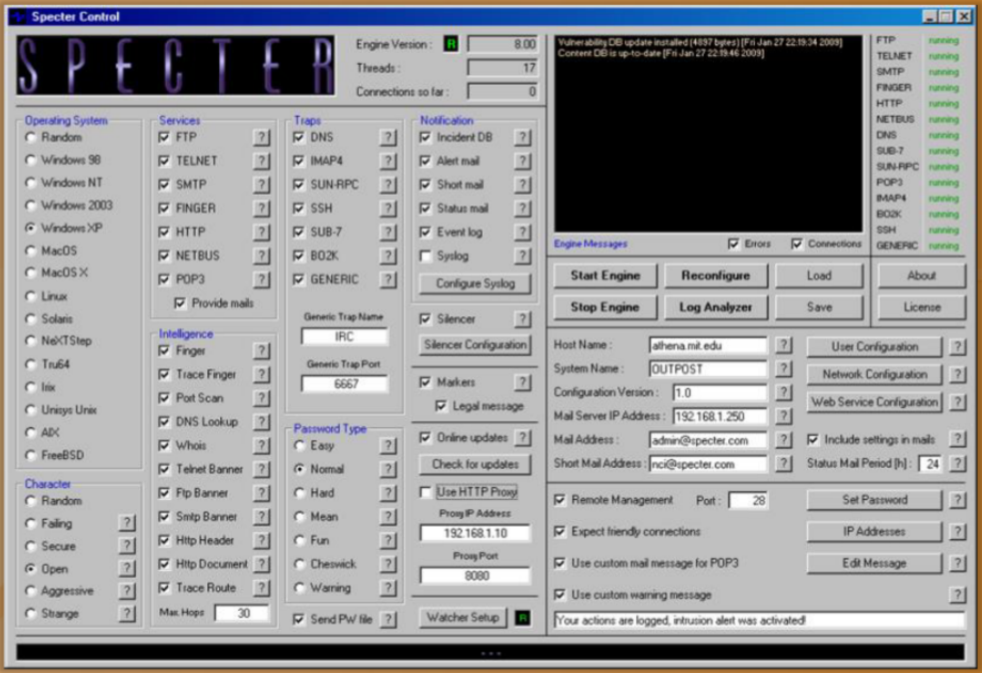
SPECTER is a **honeypot** or **deception system**. It simulates a **complete system**, providing an **interesting target** to lure hackers away from production systems. It offers common Internet services such as SMTP, FTP, POP3, HTTP, and TELNET, which appear perfectly normal to attackers. However, they are traps so that traces are left without the attacker knowing that they are connected to a decoy system that does none of the things it appears to do; but instead, it logs everything and notifies the appropriate people.

Furthermore, SPECTER automatically **investigates attackers** while they are still trying to break in. It provides massive amounts of decoy content and it generates decoy programs that can't leave hidden marks on the attacker's computer. Automated weekly online updates of the honeypot's content and vulnerability databases allow the honeypot to change constantly without user interaction.

Advantages:

- Suspicious interest in the network, and computers, can be detected immediately.
- Administrators are notified of hostile activity when it happens, so that they can immediately look at the problem and take action.

- The system is very easy to set up and configure while providing sophisticated features. Fully automated online updates of the honeypot's content and vulnerability databases allow the honeypot to change constantly without user interaction.
- There cannot be false alerts, as a legitimate user cannot connect to the honeypot.
- Specter simulates in **14 different operating systems**:
- Windows 98, Windows NT, Windows 2000, Windows XP, Linux, Solaris, Tru64, NeXTStep, Irix, Unisys Unix, AIX, MacOS, MacOS X, and FreeBSD.

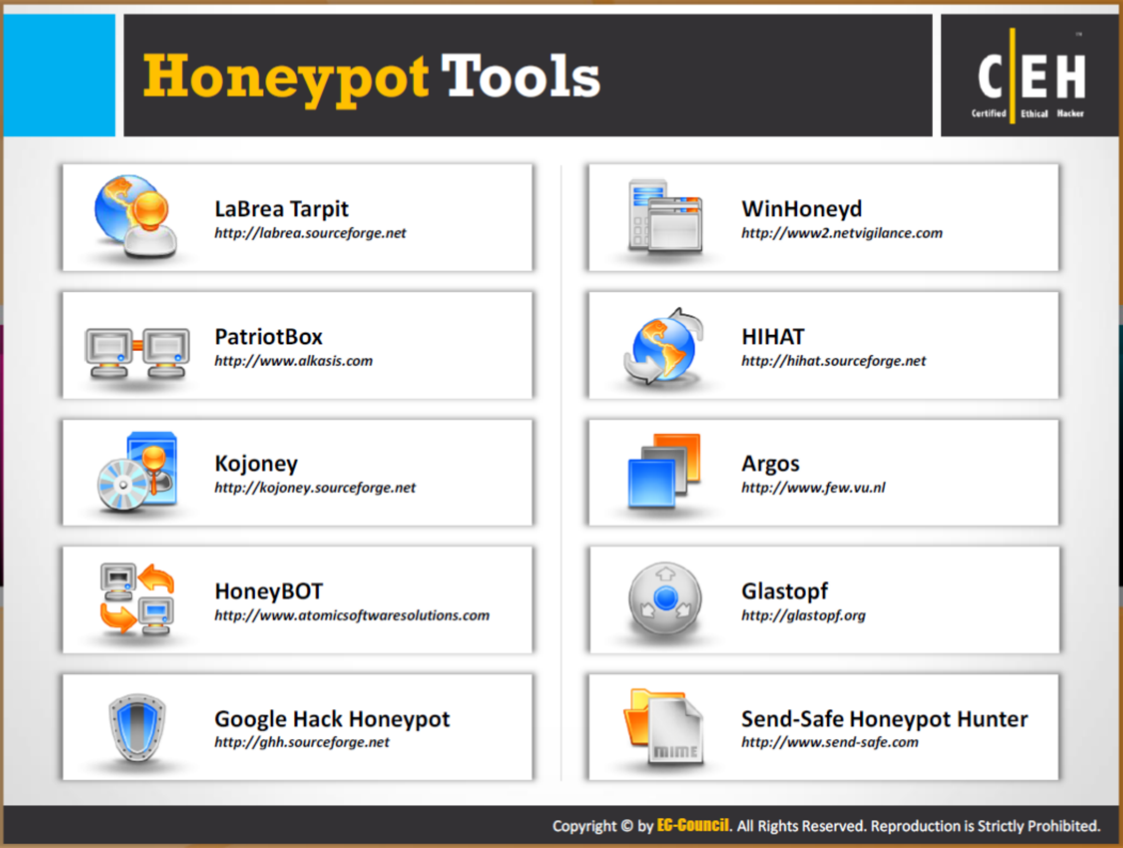

Specter Control
SPECTER
Operating System
Random
Windows 98
Windows NT
Windows 2003
Windows XP
MacOS
MacOS X
Linux
Solaris
NeXTStep
Tru64
Irix
Unisys Unix
AIX
FreeBSD
Character
Random
Failing
Secure
Open
Aggressive
Strange
Services
FTP
TELNET
SMTP
FINGER
HTTP
NETBUS
POP3
Provide mails
Intelligence
Finger
Trace Finger
Port Scan
DNS Lookup
Whois
Telnet Banner
Ftp Banner
Smtp Banner
Http Header
Http Document
Trace Route
Traps
DNS
IMAP4
SUN-RPC
SSH
SUB-7
BO2K
GENERIC
Password Type
Easy
Normal
Hard
Mean
Fun
Cheswick
Warning
Send PW file
Notification
Incident DB
Alert mail
Short mail
Status mail
Event log
Syslog
Configure Syslog
Silencer
Silencer Configuration
Markers
Legal message
Online updates
Check for updates
Use HTTP Proxy
Watcher Setup
Engine Messages
Errors
Connections
Start Engine
Reconfigure
Load
About
Stop Engine
Log Analyzer
Save
License
User Configuration
Network Configuration
Web Service Configuration
Include settings in mails
Set Password
IP Addresses
Edit Message
Remote Management
Expect friendly connections
Use custom mail message for POP3
Use custom warning message
FTP running
TELNET running
SMTP running
FINGER running
HTTP running
NETBUS running
DNS running
SUB-7 running
SUN-RPC running
POP3 running
IMAP4 running
BO2K running
SSH running
GENERIC running

Engine Version: 8.00
Threads: 17
Connections so far: 0

```
Vulnerability DB update installed (4897 bytes) [Fri Jan 27 22:19:34 2009]
Content DB is up-to-date [Fri Jan 27 22:19:46 2009]
```

Operating System: Windows XP
Character: Open
Password Type: Normal
Max. Hops: 30
Generic Trap Name: IRC
Generic Trap Port: 6667
Proxy IP Address: 192.168.1.10
Proxy Port: 8080
Host Name: athena.mit.edu
System Name: OUTPOST
Configuration Version: 1.0
Mail Server IP Address: 192.168.1.250
Mail Address: admin@specter.com
Short Mail Address: nci@specter.com
Status Mail Period [h]: 24
Remote Management Port: 28
Your actions are logged, intrusion alert was activated!

FIGURE 17.20: SPECTER Screenshot



Honeypot Tools

LaBrea Tarpit
http://labrea.sourceforge.net

PatriotBox
http://www.alkasis.com

Kojoney
http://kojoney.sourceforge.net

HoneyBOT
http://www.atomicsoftwaresolutions.com

Google Hack Honeypot
http://ghh.sourceforge.net

WinHoneyd
http://www2.netvigilance.com

HIHAT
http://hihat.sourceforge.net

Argos
http://www.few.vu.nl

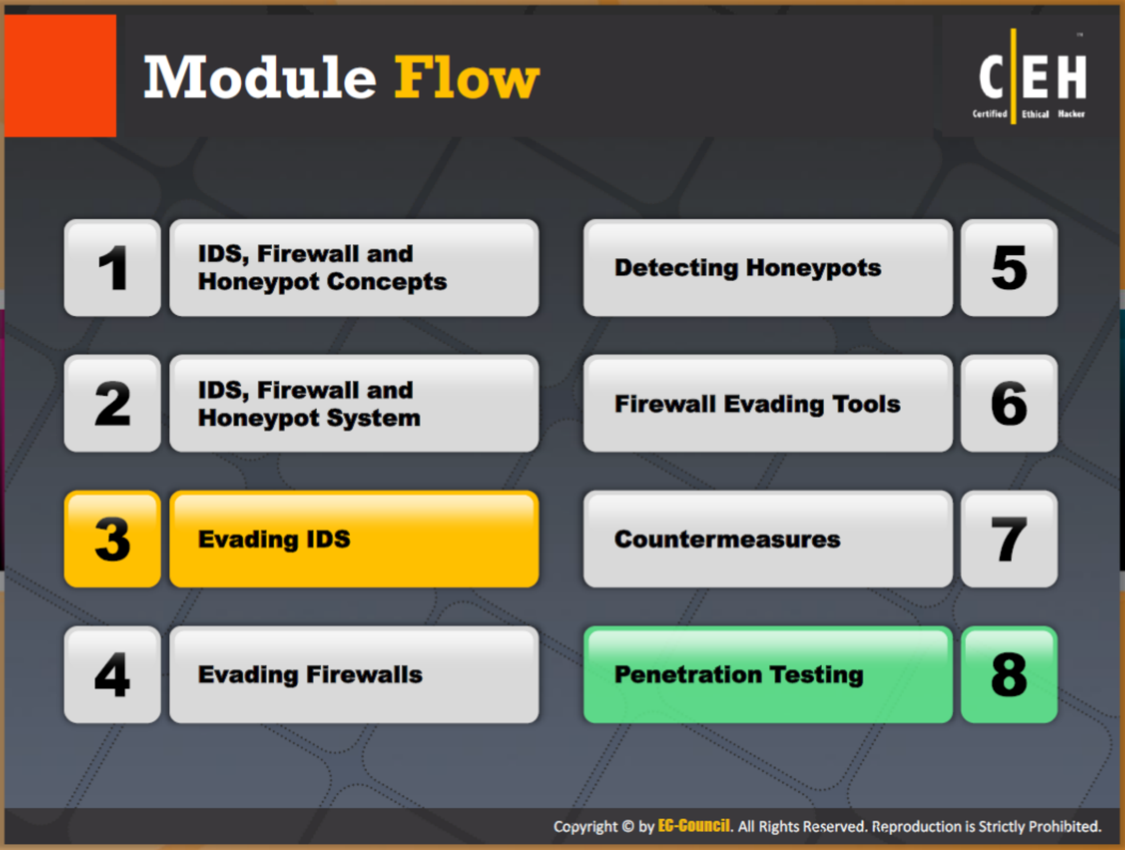
Glastopf
http://glastopf.org

Send-Safe Honeypot Hunter
http://www.send-safe.com

Honeypot Tools

Honeypots are the security tools that give the **security community** an opportunity to monitor attackers' tricks and exploits by logging their every activity, so that they can respond to these exploits quickly without attackers actually misusing and compromising systems. A few honeypot tools are listed as follows:

- LaBrea Tarpit available at http://labrea.sourceforge.net
- PatriotBox available at http://www.alkasis.com
- Kojoney available at http://kojoney.sourceforge.net
- HoneyBOT available at http://www.atomicsoftwaresolutions.com
- Google Hack Honeypot available at http://ghh.sourceforge.net
- WinHoneyd available at http://www2.netvigilance.com
- HIHAT available at http://hihat.sourceforge.net
- Argos available at http://www.few.vu.nl
- Glastopf available at http://glastopf.org
- Send-Safe Honeypot Hunter available at http://www.send-safe.com



Module Flow

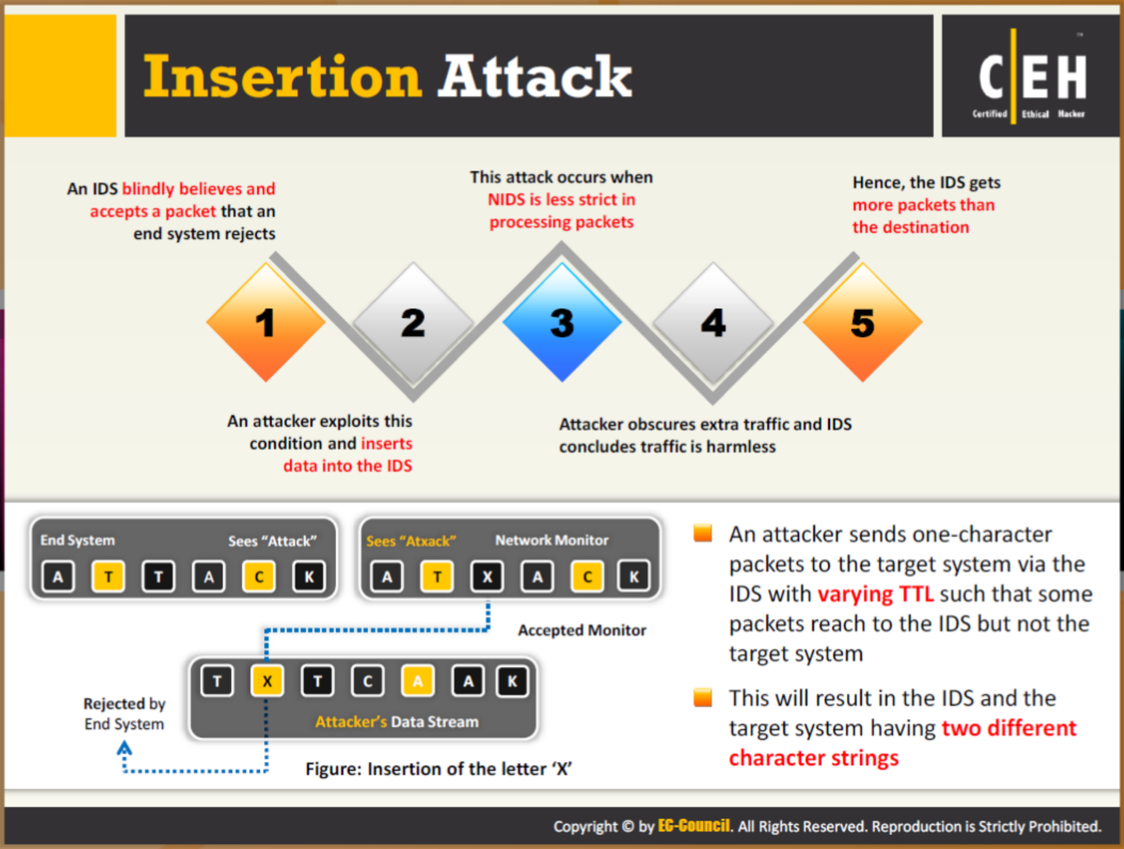
1 IDS, Firewall and Honeypot Concepts
2 IDS, Firewall and Honeypot System
3 Evading IDS
4 Evading Firewalls
5 Detecting Honeypots
6 Firewall Evading Tools
7 Countermeasures
8 Penetration Testing

Module Flow

An IDS is the critical security mechanism implemented in order to prevent intrusions and at the same time, to alert the security personnel when an attacker attempts to intrude into the network. An IDS can detect the attacker's attempts of breaking into the network. In order to avoid being detected by the IDS, attackers try to evade IDSes.

IDS, Firewall and Honeypot Concepts	**Detecting Honeypots**
IDS, Firewall and Honeypot System	**Firewall Evading Tools**
Evading IDS	**Countermeasure**
Evading Firewall	**Penetration Testing**

This section describes the ways in which attackers try to evade IDSes.



Insertion Attack

1. An IDS blindly believes and accepts a packet that an end system rejects
2. An attacker exploits this condition and inserts data into the IDS
3. This attack occurs when NIDS is less strict in processing packets
4. Attacker obscures extra traffic and IDS concludes traffic is harmless
5. Hence, the IDS gets more packets than the destination

Figure: Insertion of the letter 'X'

- An attacker sends one-character packets to the target system via the IDS with **varying TTL** such that some packets reach to the IDS but not the target system
- This will result in the IDS and the target system having **two different character strings**

Insertion Attack

The process where the **attacker confuses** the **IDS by forcing** it to read the invalid packets is known as insertion, that is, the packet would not be accepted by the system to which it is addressed. If a packet is malformed or if it does not reach its actual destination, the packet is invalid. If the IDS read an invalid packet, the IDS will become confused.

To understand how insertion becomes a problem for a network IDS, it is important to understand how **IDSes detect attacks**. The IDS employs pattern-matching algorithms to look for specific patterns of data in a packet or stream of packets. For example, IDSes might look for the string "phf" in an HTTP request to discover a PHF Common Gateway Interface (CGI) attack. An attacker who can insert packets into the IDS can prevent pattern matching from working. For instance, an attacker can send the string "phf" to a web server, attempting to exploit the CGI vulnerability, but force the IDS to read "phoneyf" (by "inserting" the string "oney") instead. One simple insertion attack involves intentionally corrupting the IP checksum. Every packet transmitted on an IP network has a checksum that is used to verify whether the packet was corrupted in transit. IP checksums are 16-bit numbers that are computed by examining information in the packet. If the checksum on an IP packet does not match the actual packet, the host to which it is addressed will not accept it, while the IDS might consider it as part of the effective stream.

For example, the attacker can send packets whose Time to live fields have been crafted to reach the IDS but not the target computers. An attacker confronts the IDS with a stream of one-character packets (the attacker-originated data stream), in which one of the characters (the letter `X') will be accepted only by the IDS. As a result, the IDS and the end system reconstruct two different strings.

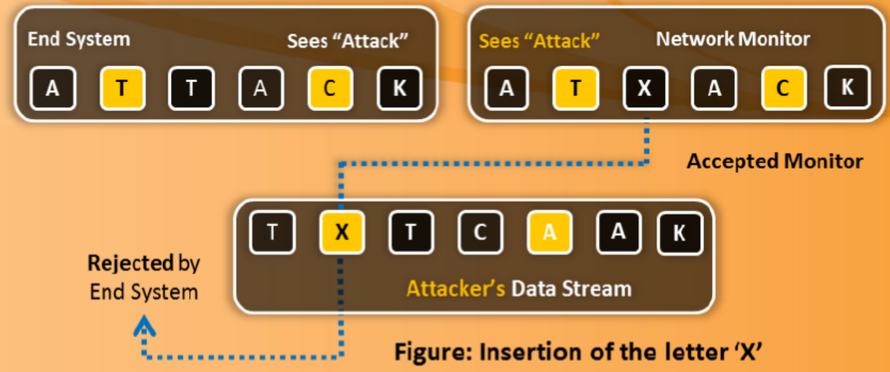
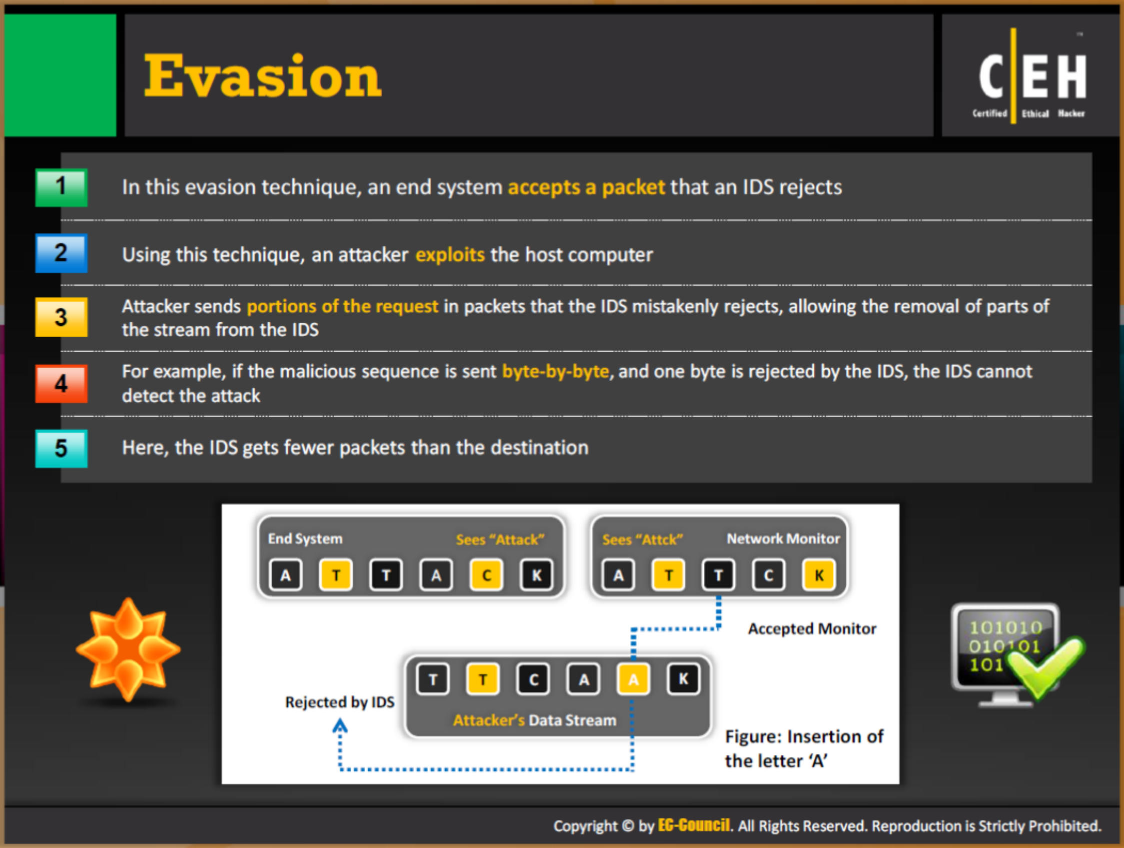

End System
Sees "Attack"
A
T
T
A
C
K
Sees "Attack"
Network Monitor
A
T
X
A
C
K
Accepted Monitor
T
X
T
C
A
A
K
Attacker's Data Stream
Rejected by
End System
Figure: Insertion of the letter 'X'

FIGURE 17.21: Insertion Attack



Evasion

1. In this evasion technique, an end system **accepts a packet** that an IDS rejects
2. Using this technique, an attacker **exploits** the host computer
3. Attacker sends **portions of the request** in packets that the IDS mistakenly rejects, allowing the removal of parts of the stream from the IDS
4. For example, if the malicious sequence is sent **byte-by-byte**, and one byte is rejected by the IDS, the IDS cannot detect the attack
5. Here, the IDS gets fewer packets than the destination

Figure: Insertion of the letter 'A'

Evasion

An "evasion" attack occurs when the **IDS discards** a packet that the host to which it is addressed accepts. Evasion attacks are devastating to the accuracy of the IDS. An evasion attack at the IP layer allows an attacker to attempt arbitrary attacks against hosts on a network, without the IDS ever realizing it. The attacker sends portions of the request in packets that the IDS mistakenly rejects, allowing the removal of parts of the stream from the **ID system's** view. For example, if the malicious sequence is sent byte-by-byte, and one byte is rejected by the IDS, the IDS cannot detect the attack. Here, the IDS gets fewer packets than the destination.

One example of an evasion attack occurs when an attacker opens a **TCP connection** with a data packet. Before any TCP connection can be used, it must be "**opened**" with a handshake between the two endpoints of the connection. A fairly obscure fact about TCP is that the handshake packets can themselves bear data. **IDSes** that do not accept the data in these packets are vulnerable to an evasion attack.

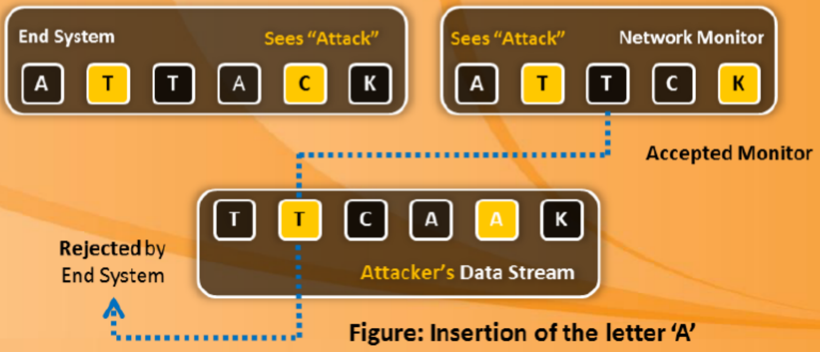
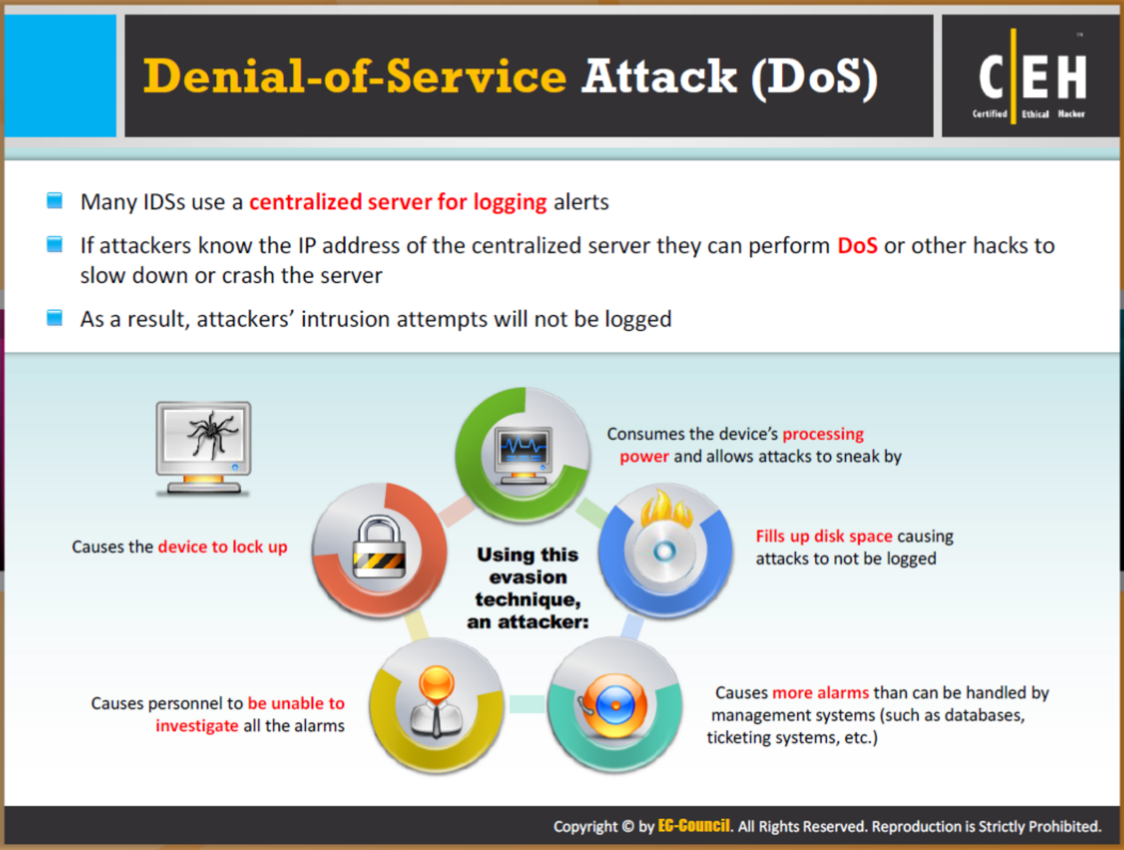

End System
Sees "Attack"
A T T A C K
Sees "Attack"
Network Monitor
A T T C K
Accepted Monitor
T T C A A K
Attacker's Data Stream
Rejected by End System
Figure: Insertion of the letter 'A'

FIGURE 17.22: Evasion



Denial-of-Service Attack (DoS)

- Many IDSs use a centralized server for logging alerts
- If attackers know the IP address of the centralized server they can perform DoS or other hacks to slow down or crash the server
- As a result, attackers' intrusion attempts will not be logged

Using this evasion technique, an attacker:

- Consumes the device's processing power and allows attacks to sneak by
- Fills up disk space causing attacks to not be logged
- Causes more alarms than can be handled by management systems (such as databases, ticketing systems, etc.)
- Causes personnel to be unable to investigate all the alarms
- Causes the device to lock up

Denial-of-Service Attack (DoS)

Multiple types of denial-of-service attacks are valid against **IDS systems**. The attacker identifies a point of network processing that requires the allocation of a resource, causing a condition to occur that consumes all of that resource. The resources that can be affected by the attacker are CPU cycles, memory, disk space, and network bandwidth. The **CPU capabilities** of the IDS can be monitored and affected. This is because IDS needs half of the CPU cycle to read the packets, detecting what the purpose of their existence is, and then comparing them with some location in the **saved network state**. An attacker can verify the most computationally expensive network processing operations and then compel the IDS to spend all its time carrying out useless work.

An IDS requires memory for a variety of things. For generating a match for the patterns, the TCP connections should be saved, the reassembly queues should be maintained, and the buffers of the data should be generated. In the initial phase, the system requires memory so that it can read the packets. Memory is allocated by the system. It is needed for network processing operations. An attacker can verify the processing operations that require the ID system to allocate memory and force the IDS to allocate all of its memory for meaningless information.

In certain circumstances, the ID systems store activity logs on the disk. The stored events occupy most of the disk space. Most computers have limited disk space. The attackers can occupy a major part of the disk space on the IDS by creating and storing a large number of useless events. This renders the **IDS useless** in terms of storing real events.

Network IDS systems record the activity on the networks they monitor. They are competent because networks are hardly ever used to their full capacity; few monitoring systems can cope with an extremely busy network.

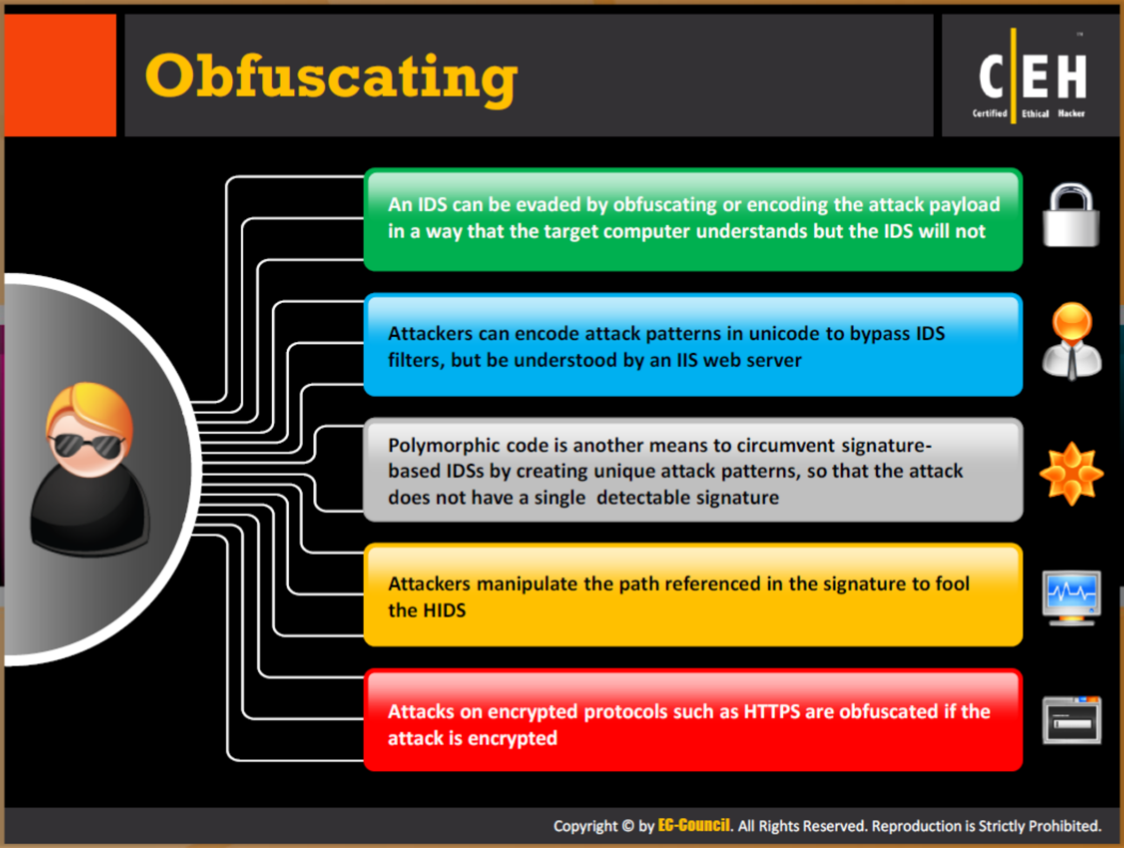
The IDS system, unlike an end system, must read everyone's packets, not just those sent specifically to it. An attacker can overload the network with meaningless information and prevent the IDS system from keeping up with what is actually happening on the network.

Many IDSes today employ central logging servers that are used exclusively to store IDS alert logs. The central server's function is to centralize alert data so it can be viewed as a whole rather than on a system-by-system basis.

However, if attackers know the **central log server's IP address**, they could slow it down or even crash it using a DoS attack. After the server is shut down, attacks could go unnoticed because the alert data is no longer being logged.

Using this evasion technique, an attacker:

- Consumes the device's processing power and allows attacks to sneak by
- Fills up disk space causing attacks to not be logged
- Causes more alarms than can be handled by management systems (such as databases, ticketing systems, etc.)
- Causes personnel to be unable to investigate all the alarms
- Causes the device to lock up



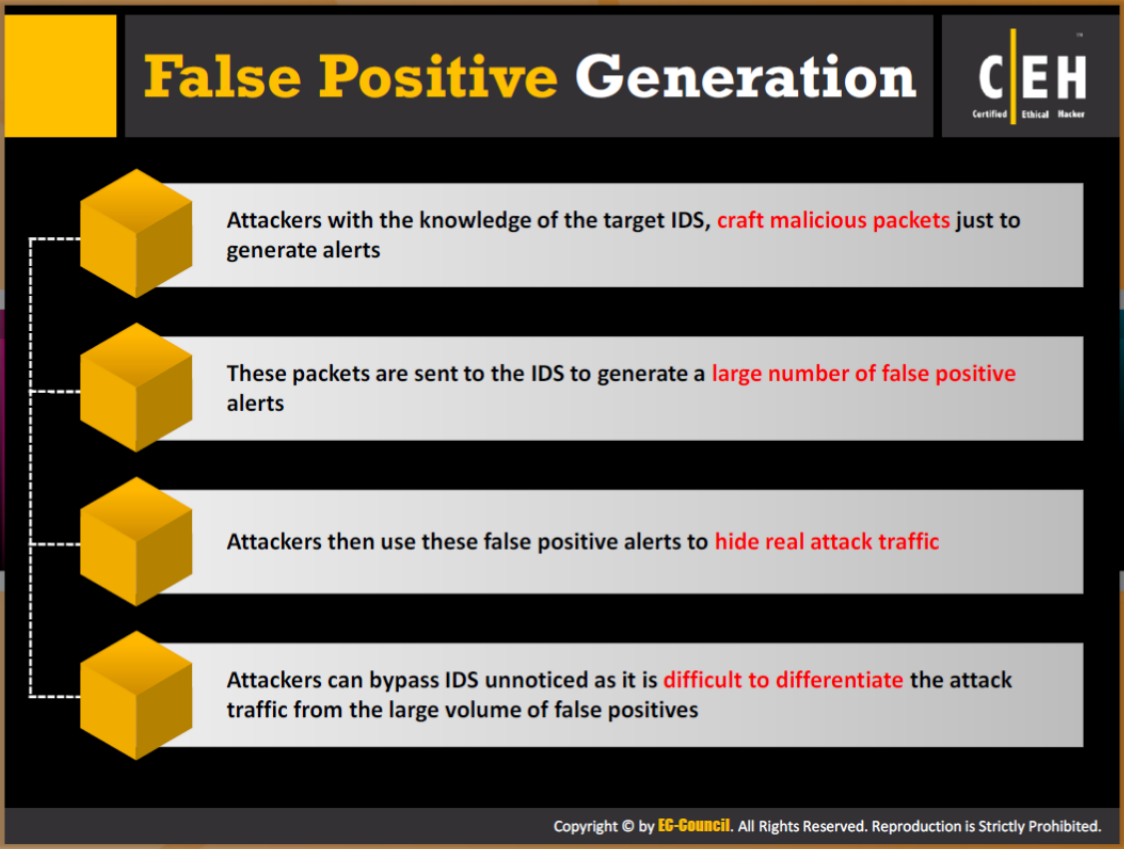
Obfuscating

- An IDS can be evaded by obfuscating or encoding the attack payload in a way that the target computer understands but the IDS will not
- Attackers can encode attack patterns in unicode to bypass IDS filters, but be understood by an IIS web server
- Polymorphic code is another means to circumvent signature-based IDSs by creating unique attack patterns, so that the attack does not have a single detectable signature
- Attackers manipulate the path referenced in the signature to fool the HIDS
- Attacks on encrypted protocols such as HTTPS are obfuscated if the attack is encrypted

Obfuscating

Obfuscation means to make code harder to understand or read, generally for privacy or security purposes. A tool called an obfuscator is sometimes used to convert a straightforward program into one that works the same way but is much harder to understand.

An IDS can be evaded by obfuscating or encoding the attack payload in a way that the target computer will reverse but the IDS will not. An attacker manipulates the path referenced in the signature to fool the HIDS. Using the Unicode character, an attacker could encode attack packets that the IDS would not recognize but that an IIS web server would decode and become attacked. Polymorphic code is another means to circumvent signature-based IDSes by creating unique attack patterns, so that the attack does not have a single detectable signature. Attacks on encrypted protocols such as HTTPS are obfuscated if the attack is encrypted. Polymorphic code is another means to circumvent signature-based IDSes by creating unique attack patterns, so that the attack does not have a single detectable signature.



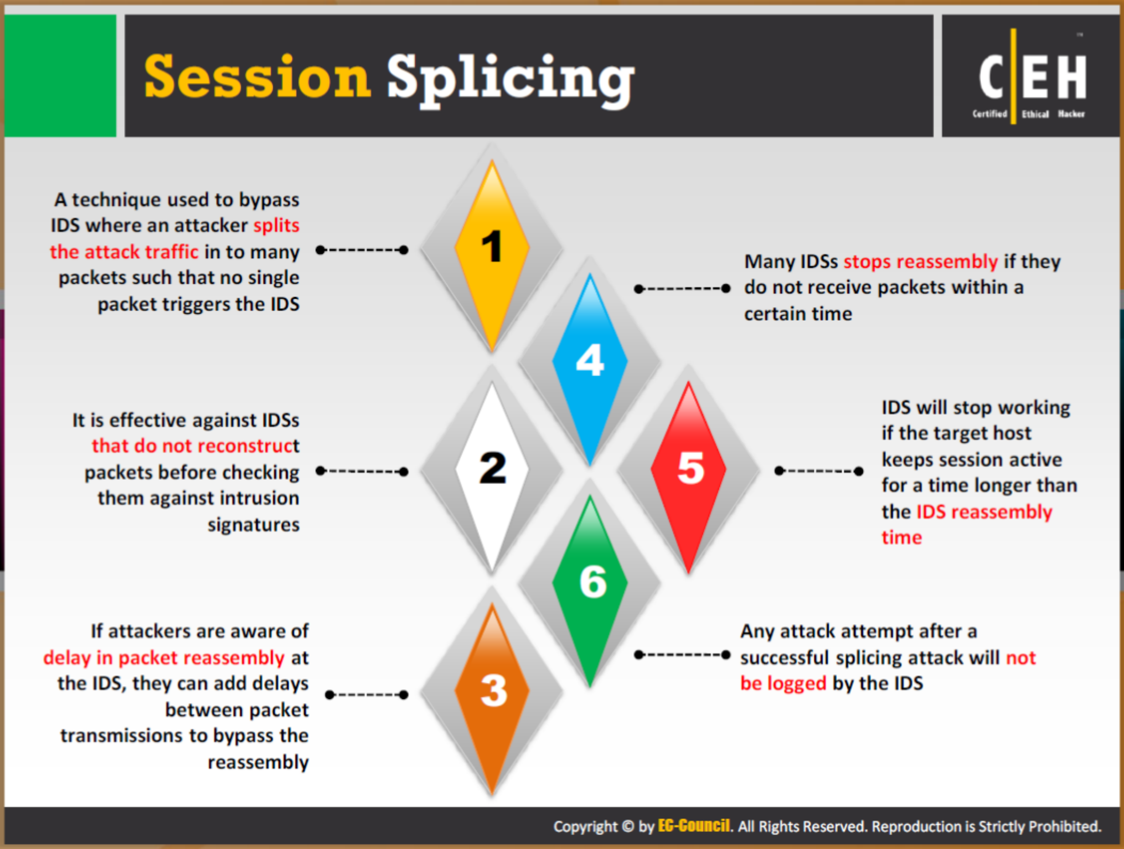
False Positive Generation

- Attackers with the knowledge of the target IDS, craft malicious packets just to generate alerts
- These packets are sent to the IDS to generate a large number of false positive alerts
- Attackers then use these false positive alerts to hide real attack traffic
- Attackers can bypass IDS unnoticed as it is difficult to differentiate the attack traffic from the large volume of false positives

False Positive Generation

This mode **does not attack** the target, but instead, it does something relatively normal. In this mode, an alarm is generated when no condition is present to warrant one. However, many IDSes falsely trigger on this.

Another attack similar to the **DoS method** is to generate a **large amount** of alert data that must be logged. Attackers craft packets known to trigger alerts within the IDS, forcing it to generate a large number of false reports. This type of attack is designed to create a great deal of log "noise" in an attempt to blend real attacks with the false. Attackers know all too well that when looking at log data, it can be very difficult to differentiate between legitimate attacks and false positives. If attackers have knowledge of the IDS system, they can even **generate false positives** specific to that IDS.

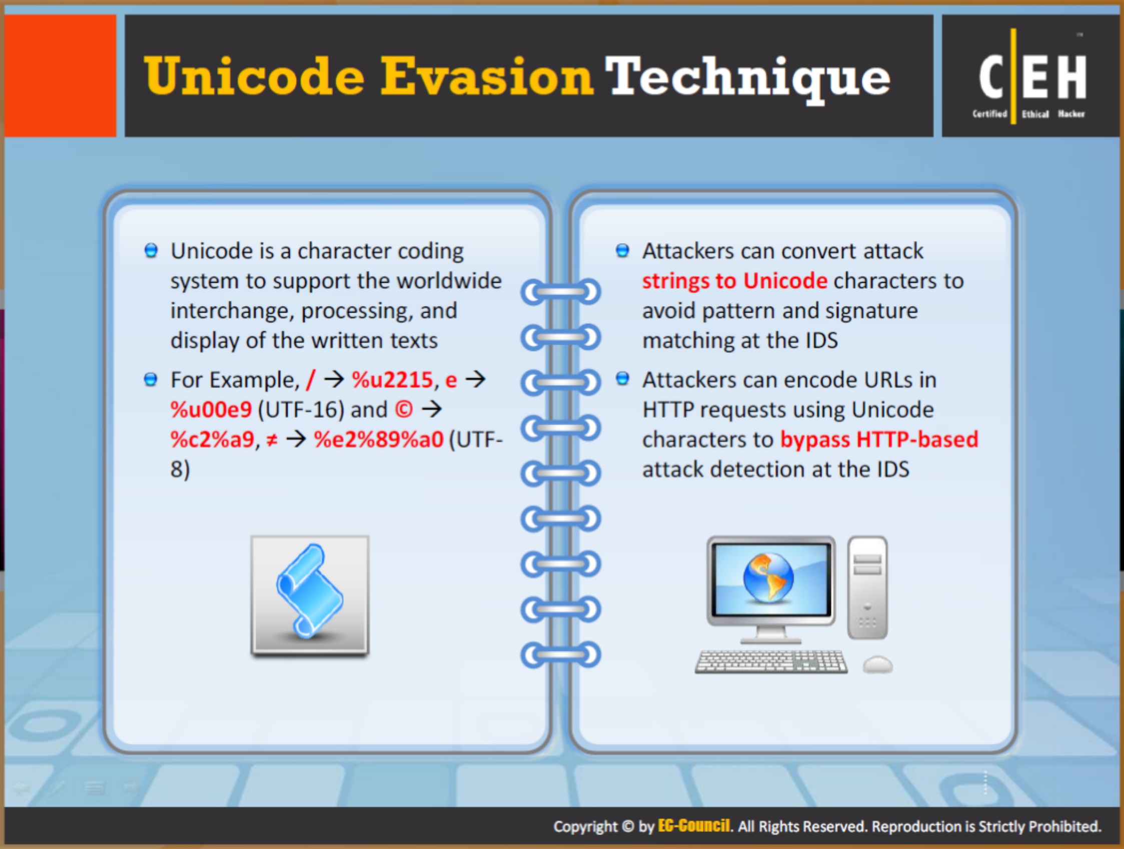


Session Splicing

1. A technique used to bypass IDS where an attacker splits the attack traffic in to many packets such that no single packet triggers the IDS
2. It is effective against IDSs that do not reconstruct packets before checking them against intrusion signatures
3. If attackers are aware of delay in packet reassembly at the IDS, they can add delays between packet transmissions to bypass the reassembly
4. Many IDSs stops reassembly if they do not receive packets within a certain time
5. IDS will stop working if the target host keeps session active for a time longer than the IDS reassembly time
6. Any attack attempt after a successful splicing attack will not be logged by the IDS

Session Splicing

Session splicing is an **IDS evasion technique** that exploits how some IDSes do not reconstruct sessions before performing pattern matching on the data. It is a network-level evasion method that divides the string across **several packets**. The data in the packets is divided into small portions of bytes and while delivering the string match is evaded. It is used by an attacker to deliver the data into several small sized packets. IDS can't handle too many small sized packets and fails to detect the **attack signatures**. If attackers know what IDS system is in use, they could add delays between packets to bypass reassembly checking. Many IDSes reassemble **communication streams**, so if a packet is not received within a reasonable amount of time, many IDSes stop reassembling and handling that stream. If the application under attack keeps a session active longer than an IDS will spend on reassembling it, the IDS will stop. As a result, any session after the IDS stops reassembling the sessions will be susceptible to malicious data theft by the attacker. Different tools such as Nessus, Whisker, etc. are used for session splicing attacks.



Unicode Evasion Technique

- Unicode is a character coding system to support the worldwide interchange, processing, and display of the written texts
- For Example, / → %u2215, e → %u00e9 (UTF-16) and © → %c2%a9, ≠ → %e2%89%a0 (UTF-8)

- Attackers can convert attack strings to Unicode characters to avoid pattern and signature matching at the IDS
- Attackers can encode URLs in HTTP requests using Unicode characters to bypass HTTP-based attack detection at the IDS

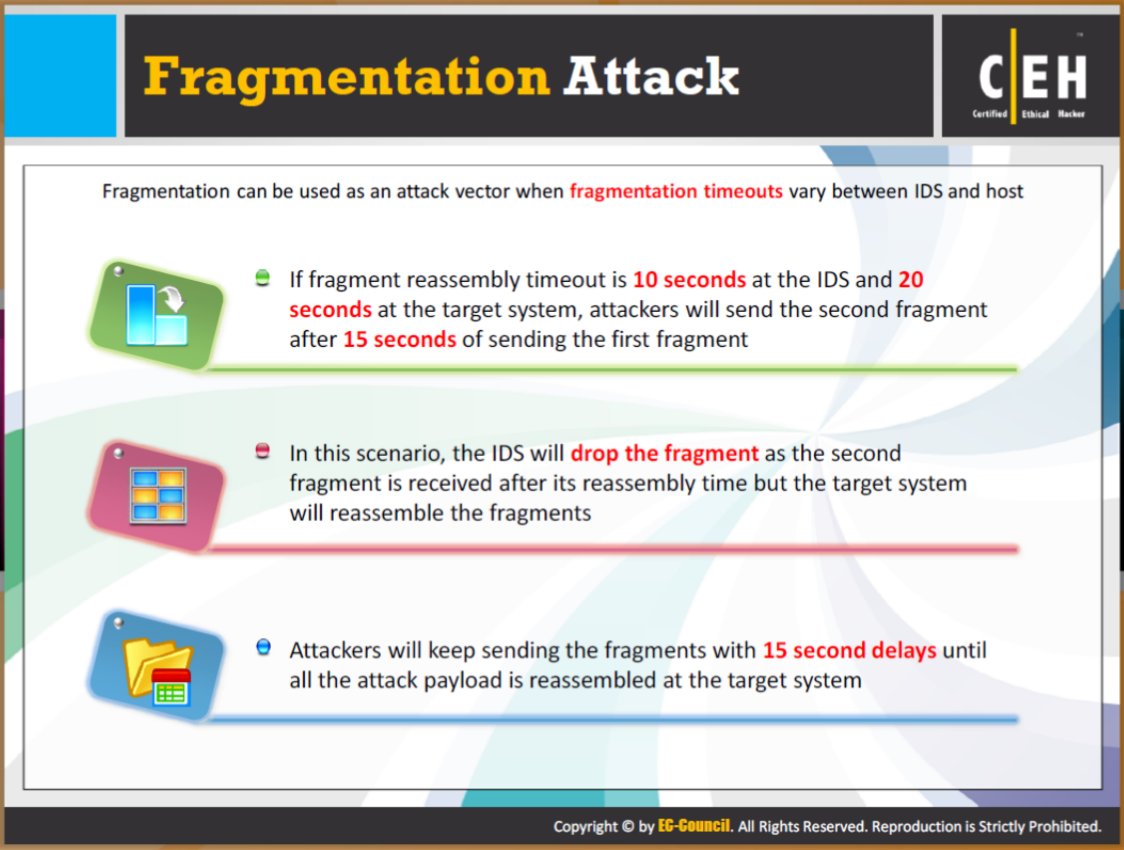
Unicode Evasion Technique

Unicode is a character representation that gives each character a unique identifier for each written language to facilitate the uniform computer representation of each language. This is problematic for IDS technology because it is possible to have multiple representations of a single character.

For example, '\' can be represented as 5C, C19C and E0819C, which makes writing pattern matching signatures very difficult.

Example for how Unicode affects IDS:

- Microsoft IIS 4.0/5.0 Directory Traversal vulnerability released in October 2000 by Rain Forrest Puppy
- This IIS vulnerability improperly restricts directory listings that were Unicode encoded within the URL request
- This allowed remote attackers to view files on the IIS server that they normally would not be permitted to see



Fragmentation Attack

Fragmentation can be used as an attack vector when fragmentation timeouts vary between IDS and host

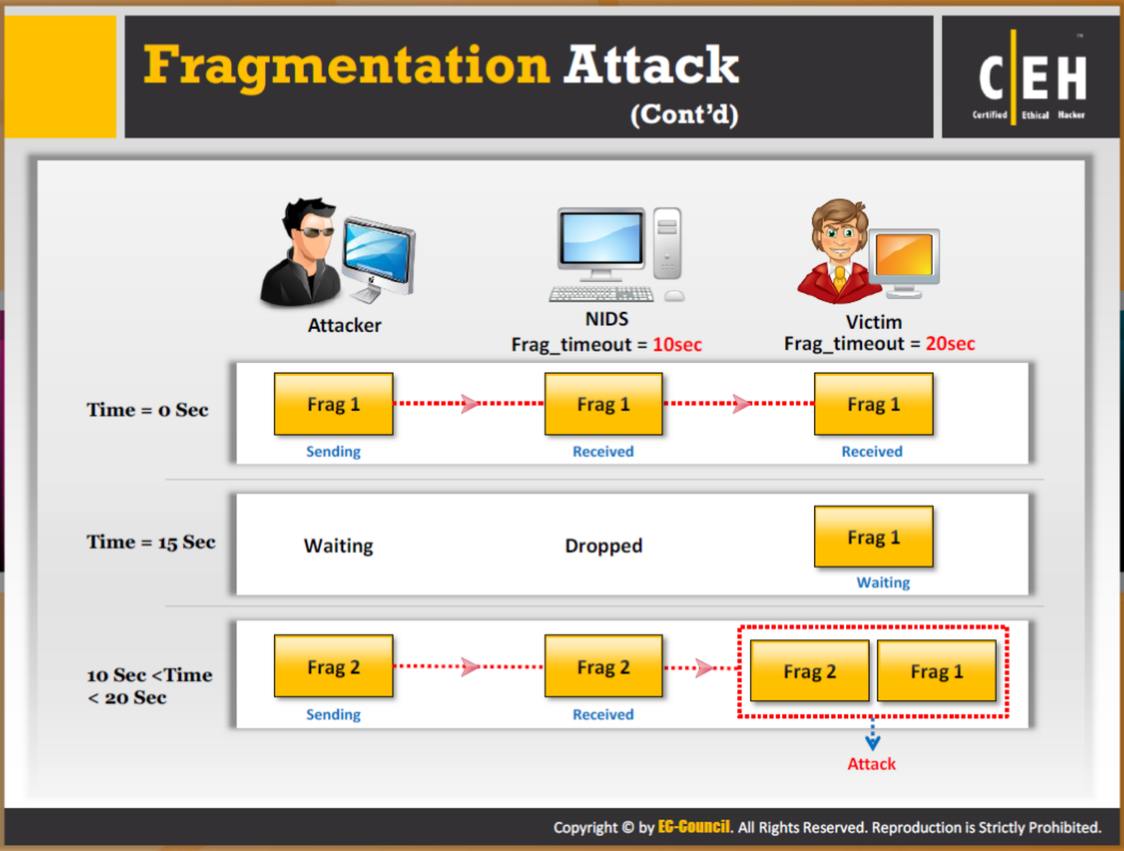
- If fragment reassembly timeout is 10 seconds at the IDS and 20 seconds at the target system, attackers will send the second fragment after 15 seconds of sending the first fragment
- In this scenario, the IDS will drop the fragment as the second fragment is received after its reassembly time but the target system will reassemble the fragments
- Attackers will keep sending the fragments with 15 second delays until all the attack payload is reassembled at the target system

Fragmentation Attack

Attackers break the single Internet protocol datagram into multiple packets of smaller size. IDS fragmentation reassembly timeout is less than fragmentation reassembly timeout of the victim.

Attack Scenario:

Assume the IDS fragmentation reassembly timeout is 15 seconds and the system is monitoring Linux hosts that have default fragmentation reassembly timeout of 30 seconds. After sending the first fragment, the attacker can send the second fragment with a delay of 15 seconds but still within 30 seconds. Now, the victim reassembles the fragments whereas at the IDS the fragmentation reassembly timeout parameter kicks in and the time out occurs. The second fragment received by the IDS will be dropped as the IDS has already lost the first fragment, due to time out. Thus, the victim will reassemble the fragments and will receive the attack whereas the IDS will not make any noise or generate alerts.

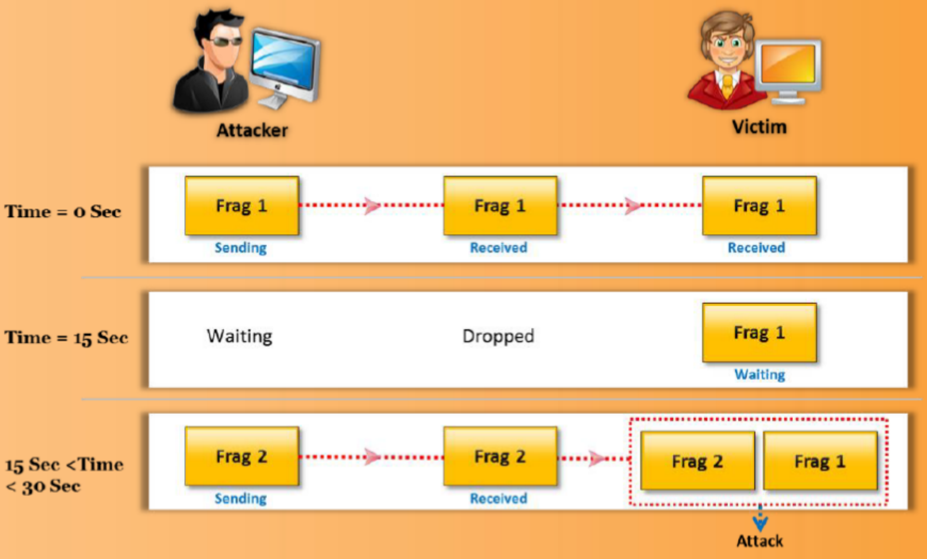


Fragmentation Attack
(Cont'd)

Attacker
NIDS
Frag_timeout = 10sec
Victim
Frag_timeout = 20sec
Time = 0 Sec
Frag 1
Sending
Frag 1
Received
Frag 1
Received
Time = 15 Sec
Waiting
Dropped
Frag 1
Waiting
10 Sec <Time < 20 Sec
Frag 2
Sending
Frag 2
Received
Frag 2
Frag 1
Attack

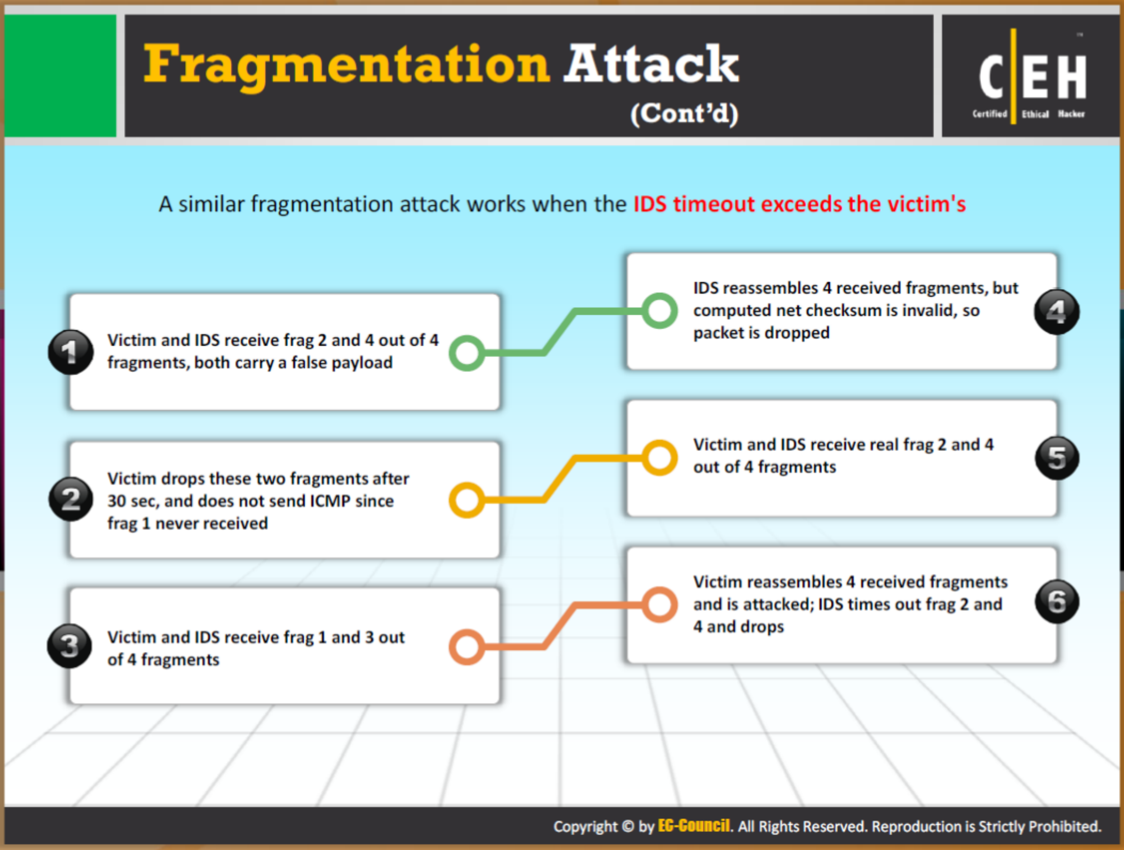
Fragmentation Attack (Cont'd)

The following figure illustrates the attack where the NIDS fragmentation re-assembly timeout is less than the victim's fragmentation reassembly timeout.



Attacker
Victim
Time = 0 Sec
Frag 1
Sending
Frag 1
Received
Frag 1
Received
Time = 15 Sec
Waiting
Dropped
Frag 1
Waiting
15 Sec <Time < 30 Sec
Frag 2
Sending
Frag 2
Received
Frag 2
Frag 1
Attack

FIGURE 17.23: NIDS Fragmentation Re-assembly Part 1



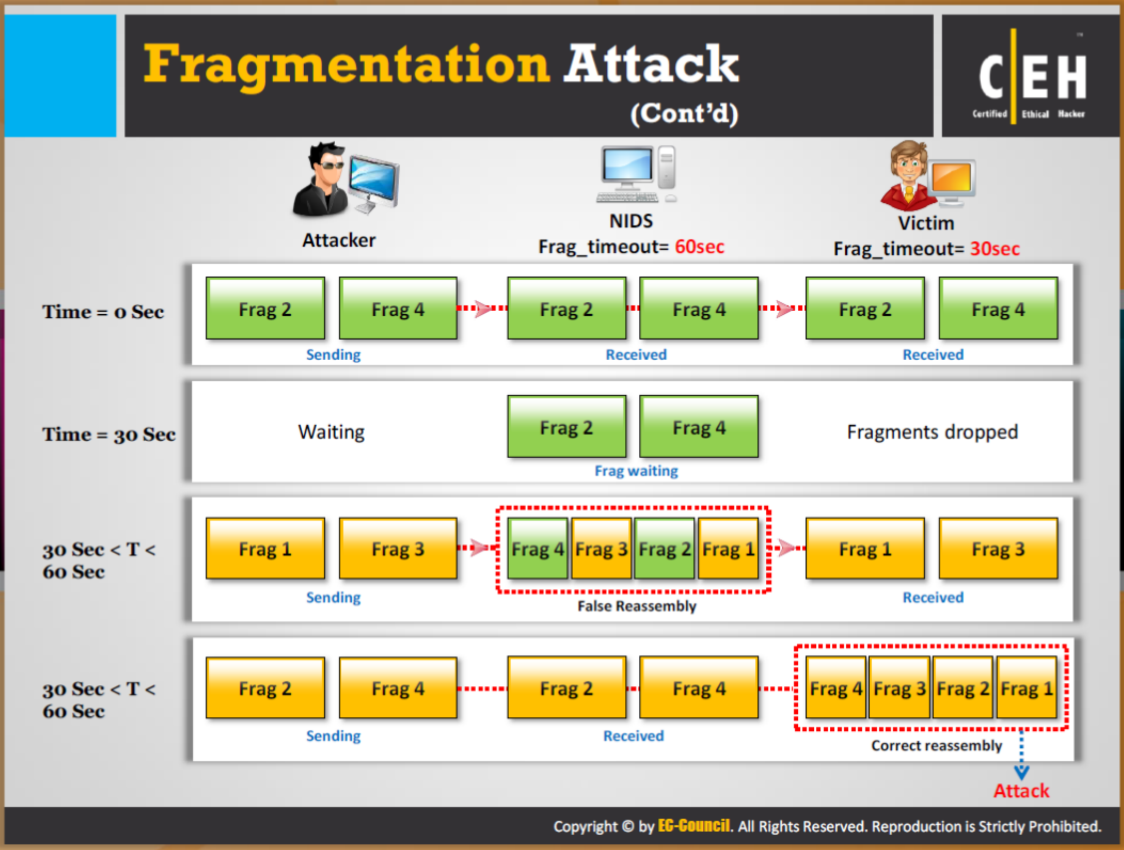
Fragmentation Attack (Cont'd)

A similar fragmentation attack works when the IDS timeout exceeds the victim's

1. Victim and IDS receive frag 2 and 4 out of 4 fragments, both carry a false payload
2. Victim drops these two fragments after 30 sec, and does not send ICMP since frag 1 never received
3. Victim and IDS receive frag 1 and 3 out of 4 fragments
4. IDS reassembles 4 received fragments, but computed net checksum is invalid, so packet is dropped
5. Victim and IDS receive real frag 2 and 4 out of 4 fragments
6. Victim reassembles 4 received fragments and is attacked; IDS times out frag 2 and 4 and drops

Fragmentation Attack (Cont'd)

An attacker has fragmented the attack packet into four segments: 1, 2, 3, and 4, and sends frag2 and frag4 with a false payload (referred as 2', 4'), which are received by both the victim and the IDS. The victim waits until the fragments' reassembly timeout occurs at the victim's end and it drops the initial fragments (30 seconds in this case). The victim still has not received fragment 1, so it will quietly drop the fragments and no ICMP error message will be thrown by the victim. The attacker then sends packets (1, 3) with legitimate payloads. At this stage, the victim has only fragments (1, 3), whereas the IDS has fragments (1, 2', 3, 4') in that 2, 4 fragments sent by attacker have a false payload. Since the IDS has all the four fragments it will do a TCP reassembly. Also, since fragments 2 and 4 have false payloads, the net checksum computed will be invalid. So, the IDS will drop the packet. If the attacker now sends fragments 2, 4 again with valid payload, the IDS will have only these two fragments, whereas the victim will have all (1, 3, 2, 4) fragments all with a valid payload, and it will do a reassembly and read the packet as an attack.



Fragmentation Attack (Cont'd)

Attacker

NIDS
Frag_timeout= 60sec

Victim
Frag_timeout= 30sec

Time = 0 Sec: Frag 2, Frag 4 (Sending) → Frag 2, Frag 4 (Received) → Frag 2, Frag 4 (Received)

Time = 30 Sec: Waiting; Frag 2, Frag 4 (Frag waiting); Fragments dropped

30 Sec < T < 60 Sec: Frag 1, Frag 3 (Sending) → Frag 4, Frag 3, Frag 2, Frag 1 (False Reassembly) → Frag 1, Frag 3 (Received)

30 Sec < T < 60 Sec: Frag 2, Frag 4 (Sending) → Frag 2, Frag 4 (Received) → Frag 4, Frag 3, Frag 2, Frag 1 (Correct reassembly) → Attack

Fragmentation Attack (Cont'd)

The following figure illustrates an attack where the NIDS fragmentation reassembly timeout is more than the victim's fragmentation reassembly timeout.

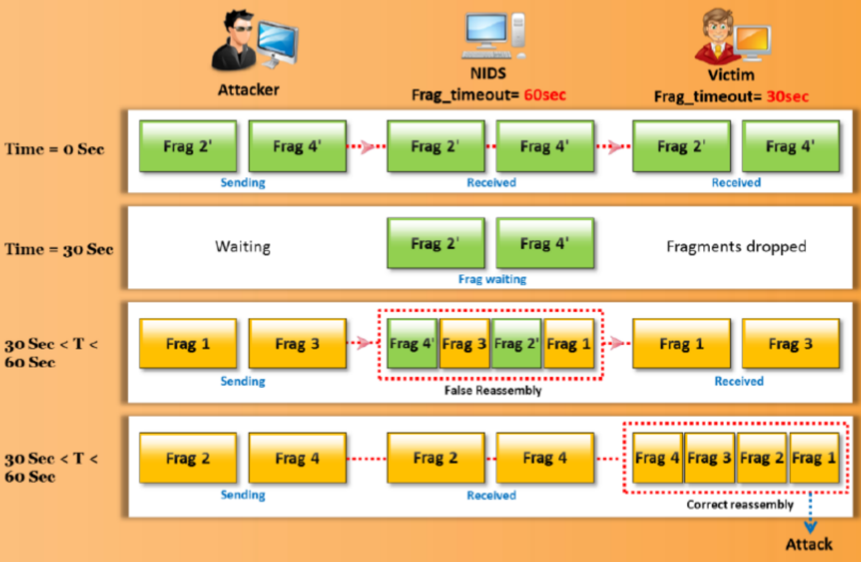
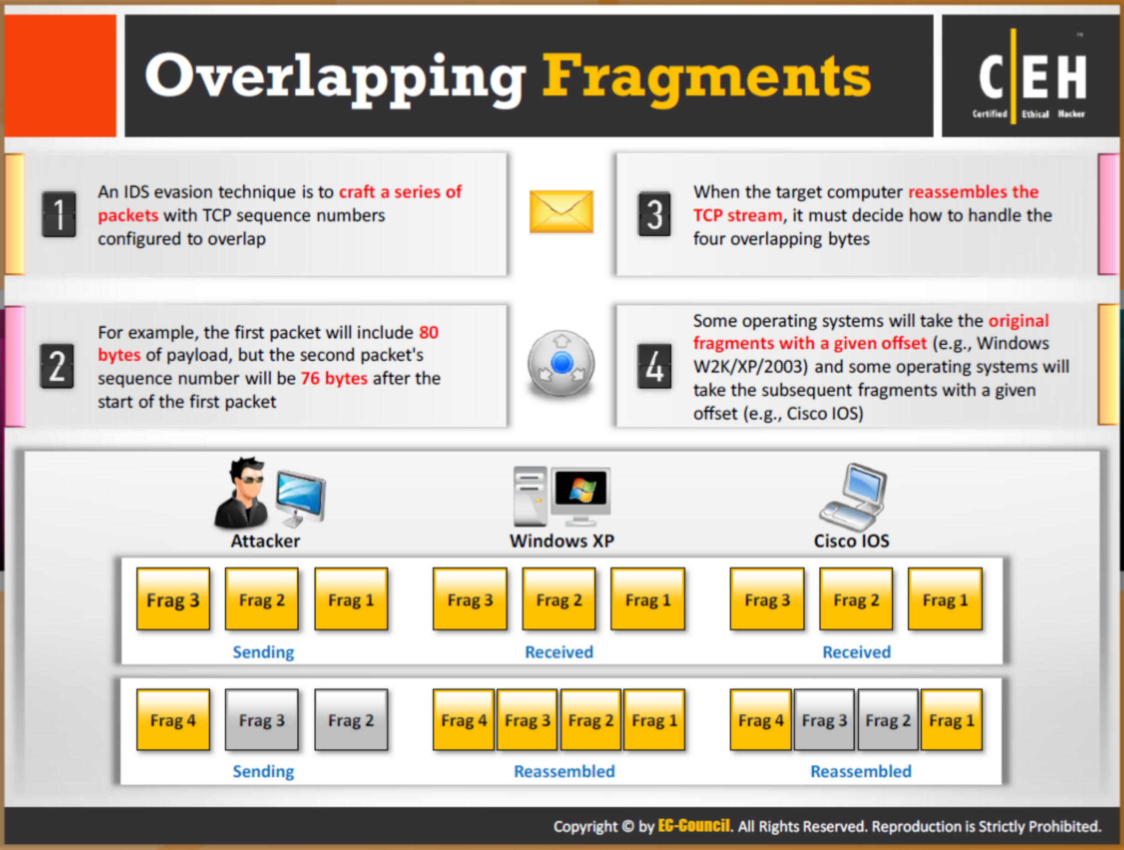

Attacker
NIDS
Frag_timeout= 60sec
Victim
Frag_timeout= 30sec
Time = 0 Sec
Frag 2'
Frag 4'
Sending
Frag 2'
Frag 4'
Received
Frag 2'
Frag 4'
Received
Time = 30 Sec
Waiting
Frag 2'
Frag 4'
Frag waiting
Fragments dropped
30 Sec < T < 60 Sec
Frag 1
Frag 3
Sending
Frag 4'
Frag 3
Frag 2'
Frag 1
False Reassembly
Frag 1
Frag 3
Received
30 Sec < T < 60 Sec
Frag 2
Frag 4
Sending
Frag 2
Frag 4
Received
Frag 4
Frag 3
Frag 2
Frag 1
Correct reassembly
Attack

FIGURE 17.24: NIDS Fragmentation Re-assembly Part 2



Overlapping Fragments

1 An IDS evasion technique is to craft a series of packets with TCP sequence numbers configured to overlap

2 For example, the first packet will include 80 bytes of payload, but the second packet's sequence number will be 76 bytes after the start of the first packet

3 When the target computer reassembles the TCP stream, it must decide how to handle the four overlapping bytes

4 Some operating systems will take the original fragments with a given offset (e.g., Windows W2K/XP/2003) and some operating systems will take the subsequent fragments with a given offset (e.g., Cisco IOS)

Attacker
Windows XP
Cisco IOS
Frag 3
Frag 2
Frag 1
Sending
Frag 3
Frag 2
Frag 1
Received
Frag 3
Frag 2
Frag 1
Received
Frag 4
Frag 3
Frag 2
Sending
Frag 4
Frag 3
Frag 2
Frag 1
Reassembled
Frag 4
Frag 3
Frag 2
Frag 1
Reassembled

Overlapping Fragments

Source: http://books.google.co.in

An IDS evasion technique is to craft a series of packets with **TCP sequence** numbers configured to overlap. In an **overlapping fragment attack**, the packets start in the middle of another packet. For example, the first packet can include **80 bytes** of payload, but the second packet's sequence number can be **76 bytes** after the start of the first packet. When the target computer reassembles the TCP stream, it must decide how to handle the four overlapping bytes. Some operating systems can take the original fragments with a given offset (e.g., Windows W2K/XP/2003) and some operating systems can take the subsequent fragments with a given offset (e.g., Cisco IOS).

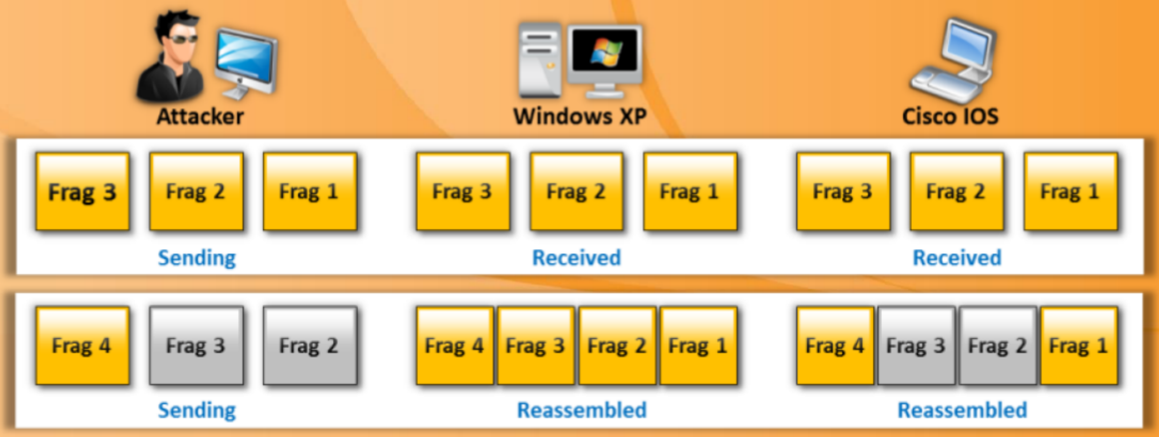
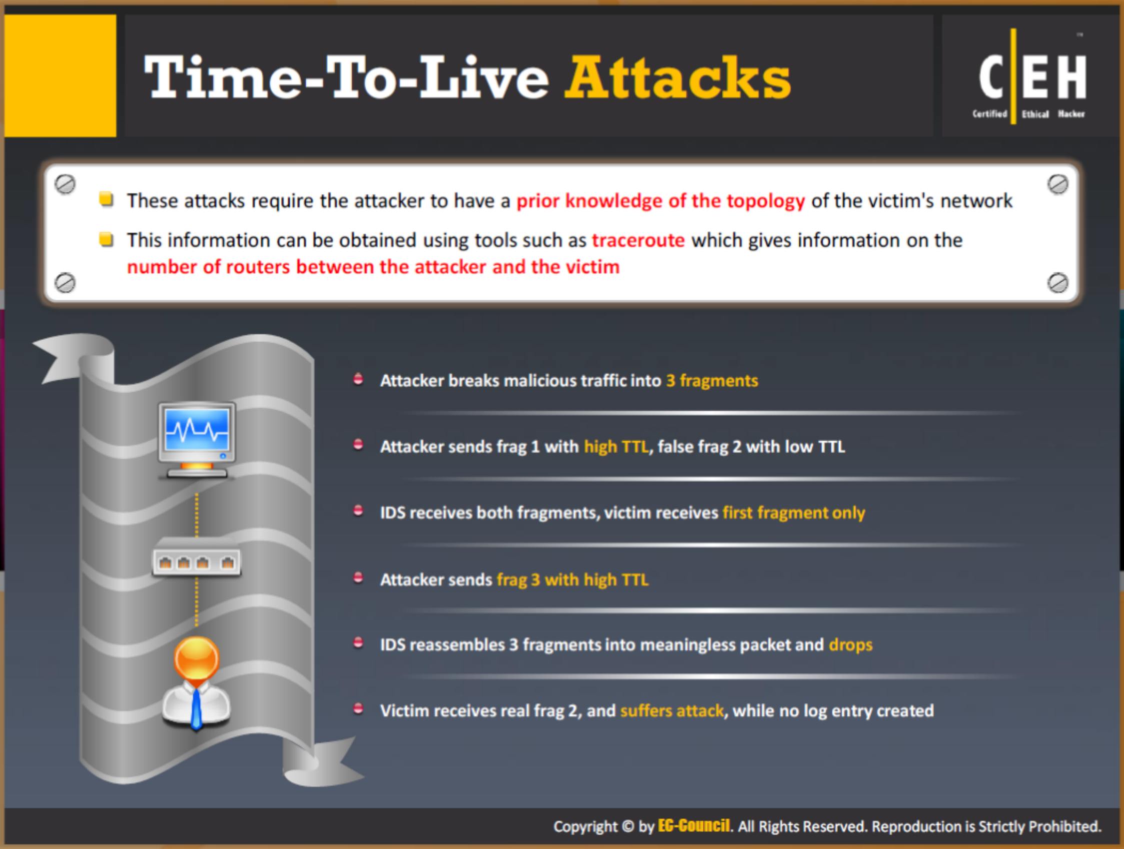

Attacker
Windows XP
Cisco IOS
Frag 3
Frag 2
Frag 1
Sending
Frag 3
Frag 2
Frag 1
Received
Frag 3
Frag 2
Frag 1
Received
Frag 4
Frag 3
Frag 2
Sending
Frag 4
Frag 3
Frag 2
Frag 1
Reassembled
Frag 4
Frag 3
Frag 2
Frag 1
Reassembled

FIGURE 17.25: Working of Overlapping Fragments



Time-To-Live Attacks

- These attacks require the attacker to have a prior knowledge of the topology of the victim's network
- This information can be obtained using tools such as traceroute which gives information on the number of routers between the attacker and the victim

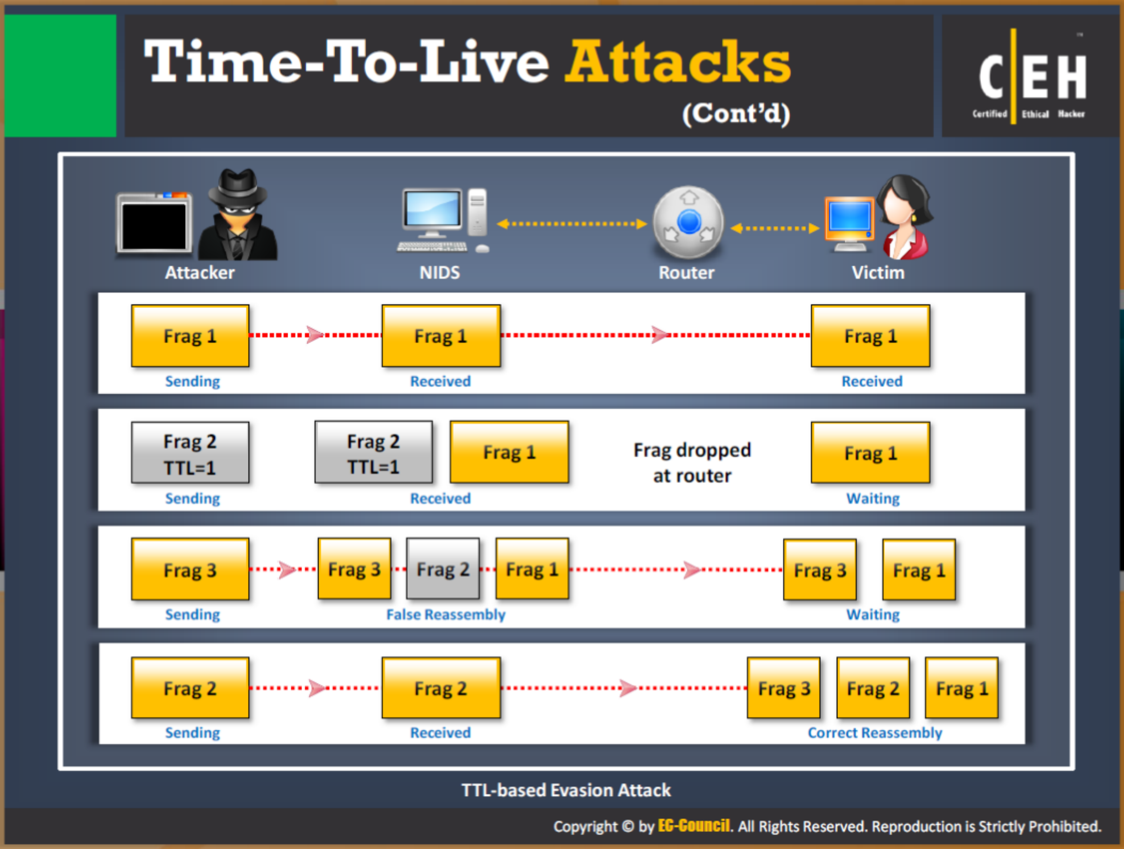
- Attacker breaks malicious traffic into 3 fragments
- Attacker sends frag 1 with high TTL, false frag 2 with low TTL
- IDS receives both fragments, victim receives first fragment only
- Attacker sends frag 3 with high TTL
- IDS reassembles 3 fragments into meaningless packet and drops
- Victim receives real frag 2, and suffers attack, while no log entry created

Time-To-Live Attacks

Source: http://www.scribd.com

Each IP packet has a field called Time to Live (TTL), which indicates how many more hops the packet should be allowed to make before being discarded or returned. Each router along a data path decrements this value, by one. When a router decrements this value to zero, it drops the packet and sends an ICMP alert notification. Typically, when a host sends a packet, it sets the TTL to a value high enough that the packet can reach its destination under normal circumstances. Different operating systems use different default initial values for the TTL. Because of this an attacker can guess the number of routers between itself and a sending machine, and make assumptions on what the initial TTL was, thereby guessing which OS a host is running, as prelude to an attack. In order to prevent such detection, SmartDefense can change the TTL field of all packets (or all outgoing packets) to a given number.

A router is present between the IDS and a victim - and the attacker is assumed to have this prior information and carries out the attack by breaking it into three fragments. Attacker sends fragment 1 with a large TTL value, which is received by both the IDS and the victim and then sends second fragment (frag2') with the TTL value of 1 and false payload. This fragment is received by the IDS, whereas the router (which is situated between the IDS and the victim) discards it as the TTL value is now reduced to zero. At this stage, the IDS has only fragment 2 as it has already performed a reassembly and the stream has been flushed. The attacker finally sends the second fragment with a valid payload and the victim performs a reassembly on fragments (1, 2, 3) and gets the attack. The attacker then sends fragment 3 with a valid TTL. This makes the IDS perform a TCP-reassembly on fragments (1, 2’, 3), whereas the victim still waits for the second fragment.



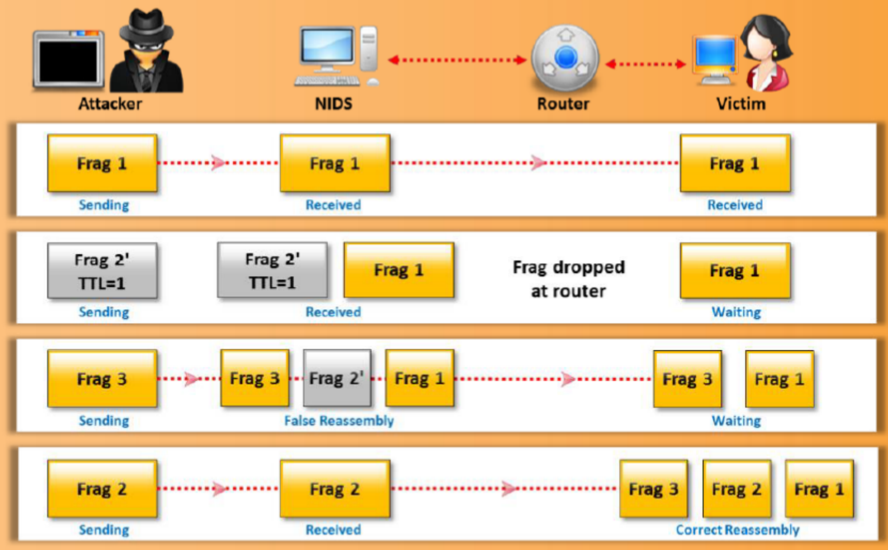
Time-To-Live Attacks (Cont'd)

C|EH
Certified Ethical Hacker
Attacker
NIDS
Router
Victim
Frag 1
Sending
Frag 1
Received
Frag 1
Received
Frag 2
TTL=1
Sending
Frag 2
TTL=1
Frag 1
Received
Frag dropped at router
Frag 1
Waiting
Frag 3
Sending
Frag 3
Frag 2
Frag 1
False Reassembly
Frag 3
Frag 1
Waiting
Frag 2
Sending
Frag 2
Received
Frag 3
Frag 2
Frag 1
Correct Reassembly

TTL-based Evasion Attack

Time-To-Live Attacks (Cont'd)

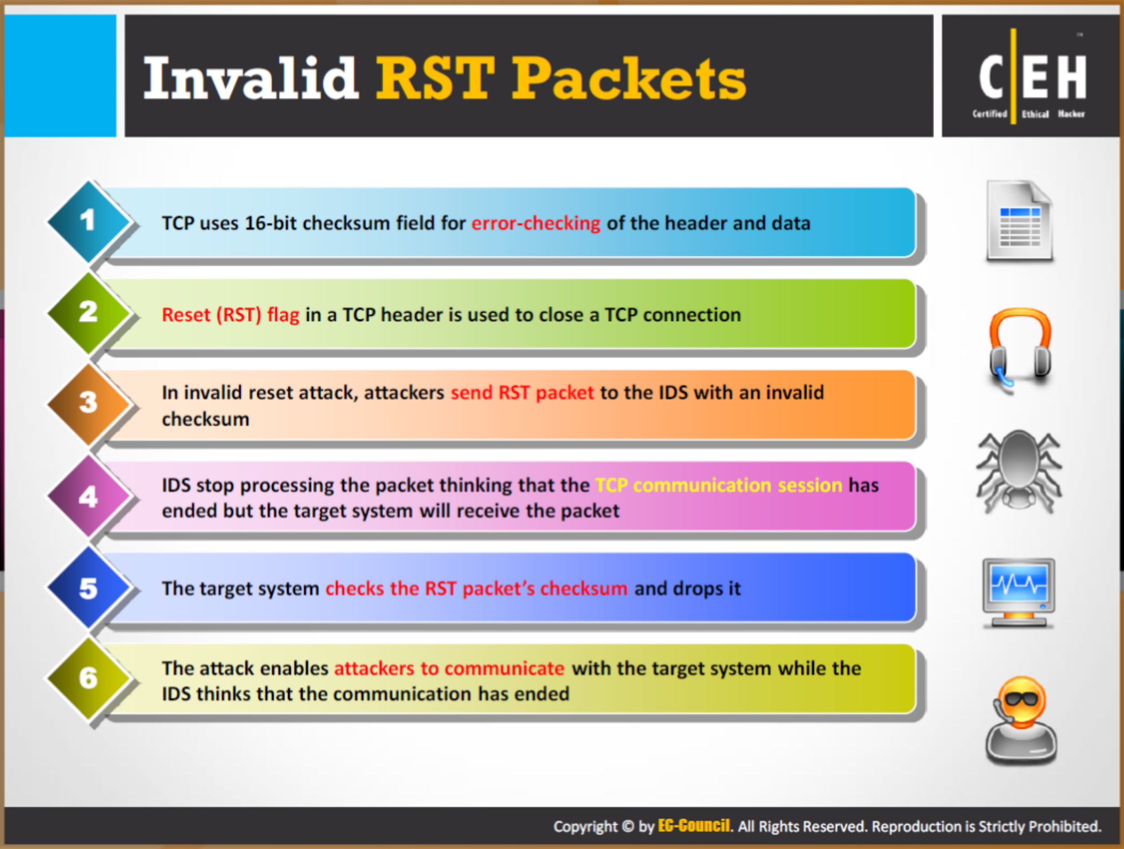
The following figure illustrates the Time-to-Live attack, a TTL-based evasion attack:



Attacker
NIDS
Router
Victim
Frag 1
Sending
Frag 1
Received
Frag 1
Received
Frag 2'
TTL=1
Sending
Frag 2'
TTL=1
Frag 1
Received
Frag dropped at router
Frag 1
Waiting
Frag 3
Sending
Frag 3
Frag 2'
Frag 1
False Reassembly
Frag 3
Frag 1
Waiting
Frag 2
Sending
Frag 2
Received
Frag 3
Frag 2
Frag 1
Correct Reassembly

TTL-based evasion attack

FIGURE 17.26: Time-To-Live Attacks



Invalid RST Packets

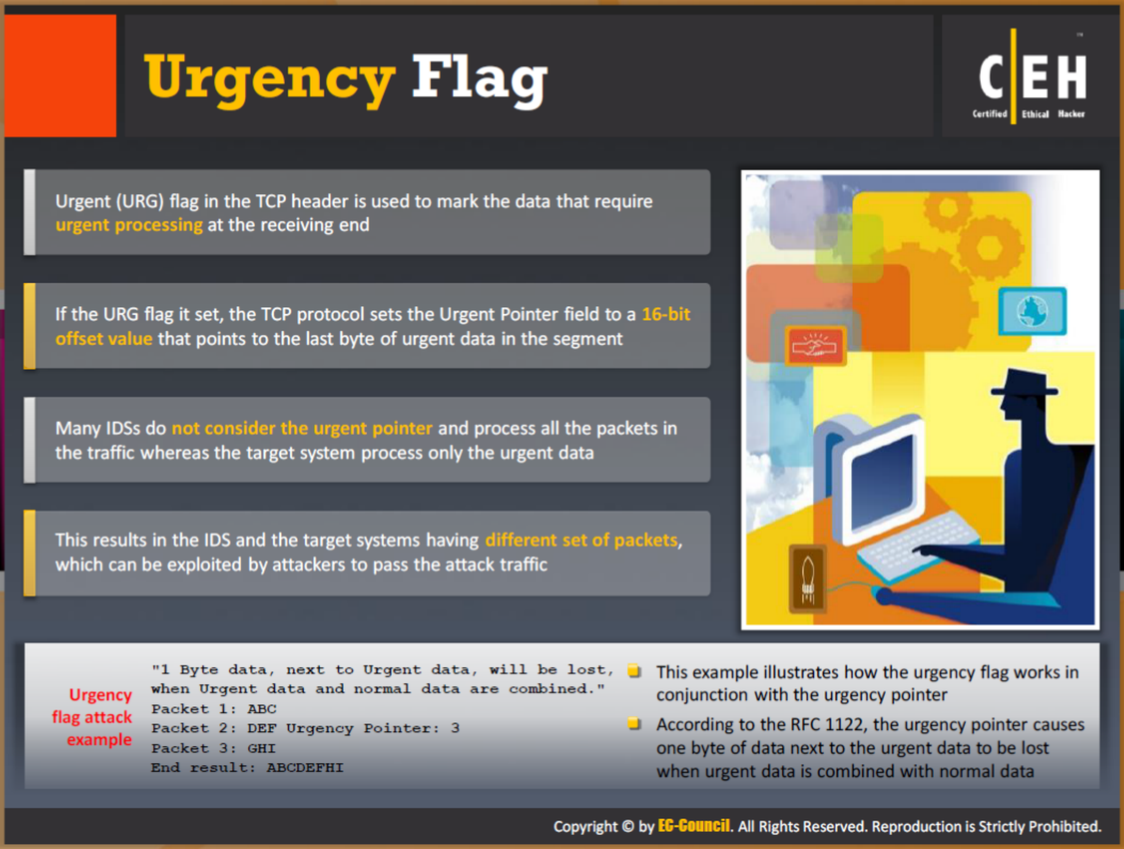
CEH
Certified Ethical Hacker

1. TCP uses 16-bit checksum field for error-checking of the header and data
2. Reset (RST) flag in a TCP header is used to close a TCP connection
3. In invalid reset attack, attackers send RST packet to the IDS with an invalid checksum
4. IDS stop processing the packet thinking that the TCP communication session has ended but the target system will receive the packet
5. The target system checks the RST packet's checksum and drops it
6. The attack enables attackers to communicate with the target system while the IDS thinks that the communication has ended

Invalid RST Packets

The **TCP protocol** uses **checksums** to ensure that communication is reliable. A checksum is added to every **transmitted segment** and it is checked at the receiving end. When a checksum differs from the checksum expected by the receiving host, the packet is dropped at the receiver's end. The TCP protocol also uses an **RST packet** to end two-way communications. Attackers can use this feature to elude detection by sending RST packets with an invalid checksum, which causes the IDS to stop processing the stream because the **IDS** thinks the communication session has ended. However, the end host sees this packet and verifies the checksum value, then drops the packet if it is invalid

Some IDS systems might interpret this packet as an actual termination of the communication and stop reassembling the communication. Such instances allow attackers to continue to communicate with the end host while confusing the IDS because the end host accepts the packets that follow the RST packet with an invalid checksum value.



Urgency Flag

C|EH
Certified Ethical Hacker

Urgent (URG) flag in the TCP header is used to mark the data that require urgent processing at the receiving end

If the URG flag it set, the TCP protocol sets the Urgent Pointer field to a 16-bit offset value that points to the last byte of urgent data in the segment

Many IDSs do not consider the urgent pointer and process all the packets in the traffic whereas the target system process only the urgent data

This results in the IDS and the target systems having different set of packets, which can be exploited by attackers to pass the attack traffic

Urgency flag attack example

```
"1 Byte data, next to Urgent data, will be lost,
when Urgent data and normal data are combined."
Packet 1: ABC
Packet 2: DEF Urgency Pointer: 3
Packet 3: GHI
End result: ABCDEFHI
```

- This example illustrates how the urgency flag works in conjunction with the urgency pointer
- According to the RFC 1122, the urgency pointer causes one byte of data next to the urgent data to be lost when urgent data is combined with normal data

Urgency Flag

The urgency flag is used within the TCP protocol to mark data as urgent. TCP uses an urgency pointer. That points to the beginning of urgent data within a packet. When the urgency flag is set, all data before the urgency pointer is ignored, and the data to which the urgency pointer points is processed. Some IDSes do not take into account the TCP protocol's urgency feature, which could allow attackers to evade the IDS, as seen in other evasion techniques. Attackers can place garbage data before the urgency. The pointer and the IDS read that data without consideration for the end host's urgency flag handling. This means the IDSes have more data than the end host actually processed.

Urgency flag attack example:

"1 Byte data, next to Urgent data, can be lost, when Urgent data and normal data are combined."

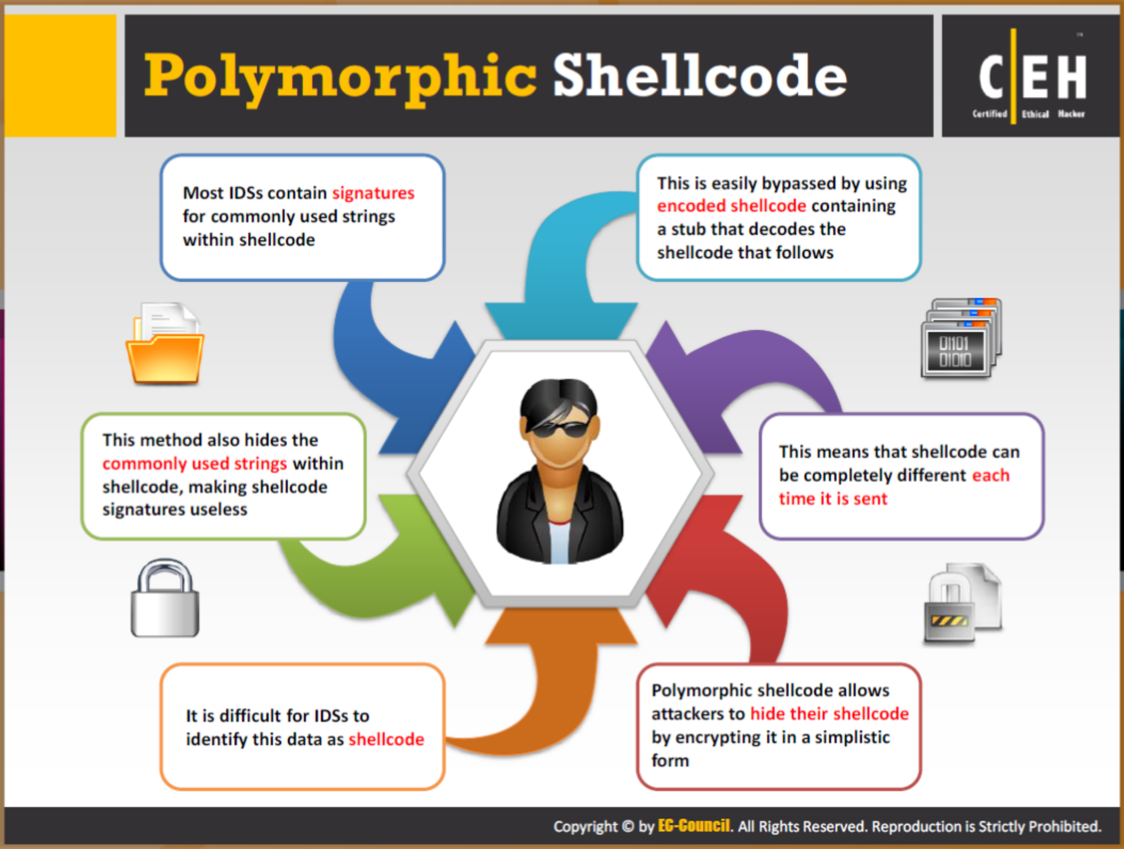
Packet 1: ABC

Packet 2: DEF Urgency Pointer: 3

Packet 3: GHI

End result: ABCDEFHI

This example illustrates how the urgency flag works in conjunction with the urgency pointer. According to the **1122 RFC**, the urgency pointer causes one byte of data next to the urgent data to be lost when urgent data is combined with normal data.



Polymorphic Shellcode

Most IDSs contain signatures for commonly used strings within shellcode

This is easily bypassed by using encoded shellcode containing a stub that decodes the shellcode that follows

This means that shellcode can be completely different each time it is sent

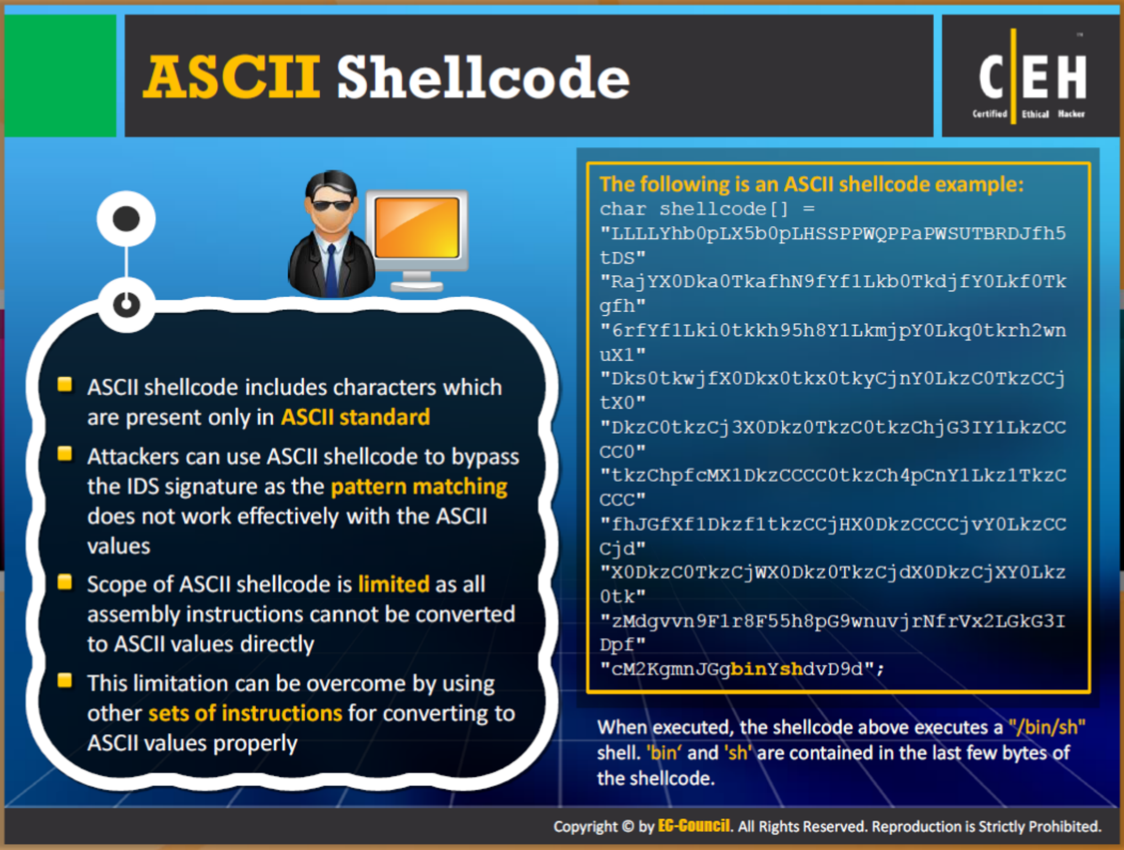
Polymorphic shellcode allows attackers to hide their shellcode by encrypting it in a simplistic form

It is difficult for IDSs to identify this data as shellcode

This method also hides the commonly used strings within shellcode, making shellcode signatures useless

Polymorphic Shellcode

Most IDSes contain **signatures** for commonly used strings within shellcode. This is easily bypassed by using **encoded shellcode** containing a stub that decodes the shellcode that follows. This means that shellcode can be completely different each time it is sent. Polymorphic shellcode allows attackers to **hide their shellcode** by encrypting it in a simplistic form. It is difficult for IDSs to identify this data as shellcode. This method also hides the commonly used strings within shellcode, making shellcode signatures useless.



ASCII Shellcode

- ASCII shellcode includes characters which are present only in **ASCII standard**
- Attackers can use ASCII shellcode to bypass the IDS signature as the **pattern matching** does not work effectively with the ASCII values
- Scope of ASCII shellcode is **limited** as all assembly instructions cannot be converted to ASCII values directly
- This limitation can be overcome by using other **sets of instructions** for converting to ASCII values properly

The following is an ASCII shellcode example:

```
char shellcode[] =
"LLLLYhb0pLX5b0pLHSSPPWQPPaPWSUTBRDJfh5
tDS"
"RajYX0Dka0TkafhN9fYf1Lkb0TkdjfY0Lkf0Tk
gfh"
"6rfYf1Lki0tkkh95h8Y1LkmjpY0Lkq0tkrh2wn
uX1"
"Dks0tkwjfX0Dkx0tkx0tkyCjnY0LkzC0TkzCCj
tX0"
"DkzC0tkzCj3X0Dkz0TkzC0tkzChjG3IY1LkzCC
CC0"
"tkzChpfcMX1DkzCCCC0tkzCh4pCnY1Lkz1TkzC
CCC"
"fhJGfXf1Dkzf1tkzCCjHX0DkzCCCCjvY0LkzCC
Cjd"
"X0DkzC0TkzCjWX0Dkz0TkzCjdX0DkzCjXY0Lkz
0tk"
"zMdgvvn9F1r8F55h8pG9wnuvjrNfrVx2LGkG3I
Dpf"
"cM2KgmnJGgbinYshdvD9d";
```

When executed, the shellcode above executes a "/bin/sh" shell. 'bin' and 'sh' are contained in the last few bytes of the shellcode.

ASCII Shellcode

ASCII shellcode contains only characters contained within the ASCII standard. This form of shellcode allows attackers to bypass commonly enforced character restrictions within string input code. It also helps attackers bypass IDS pattern matching signatures because strings are hidden within the shellcode in a similar fashion to polymorphic shellcode.

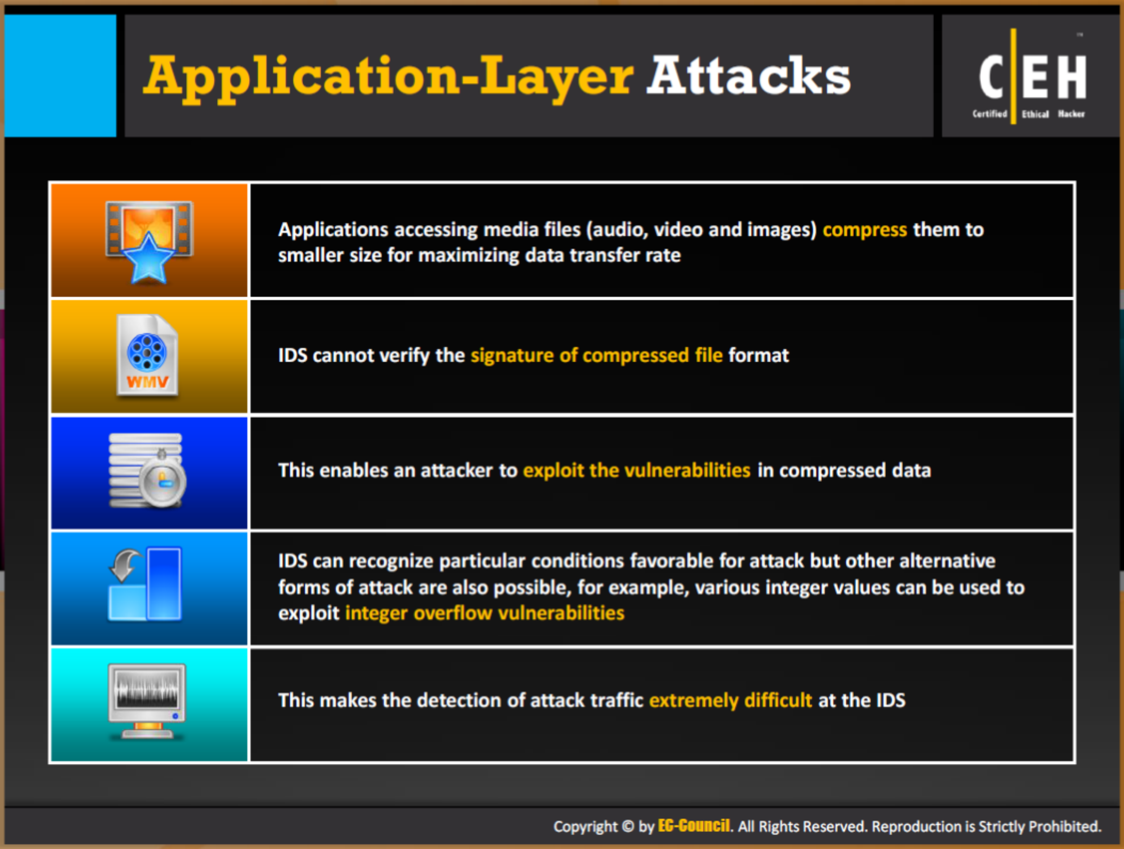
Using ASCII for shellcode is very restrictive in that it limits what the shellcode can do under some circumstances because not all assembly instructions convert directly to ASCII values. This restriction can be bypassed using other instructions or a combination of instructions that convert to ASCII character representation, which serves the same purpose of the instructions that improperly convert.

The following is an ASCII shellcode example:

```
char shellcode[] =
"LLLLYhb0pLX5b0pLHSSPPWQPPaPWSUTBRDJfh5tDS"
"RajYX0Dka0TkafhN9fYf1Lkb0TkdjfY0Lkf0Tkgfh"
"6rfYf1Lki0tkkh95h8Y1LkmjpY0Lkq0tkrh2wnuX1"
"Dks0tkwjfX0Dkx0tkx0tkyCjnY0LkzC0TkzCCjtX0"
"DkzC0tkzCj3X0Dkz0TkzC0tkzChjG3IY1LkzCCCC0"
```

```
"tkzChpfcMX1DkzCCCC0tkzCh4pCnY1Lkz1TkzCCCC"
"fhJGfXf1Dkzf1tkzCCjHX0DkzCCCCjvY0LkzCCCjd"
"X0DkzC0TkzCjWX0Dkz0TkzCjdX0DkzCjXY0Lkz0tk"
"zMdgvvn9F1r8F55h8pG9wnuvjrNfrVx2LGkG3IDpf"
"cM2KgmnJGgbinYshdvD9d";
```

When executed, the shellcode above executes a "/bin/sh" shell. 'bin' and 'sh' are contained in the last few bytes of the shellcode.



Application-Layer Attacks

Applications accessing media files (audio, video and images) compress them to smaller size for maximizing data transfer rate

IDS cannot verify the signature of compressed file format

This enables an attacker to exploit the vulnerabilities in compressed data

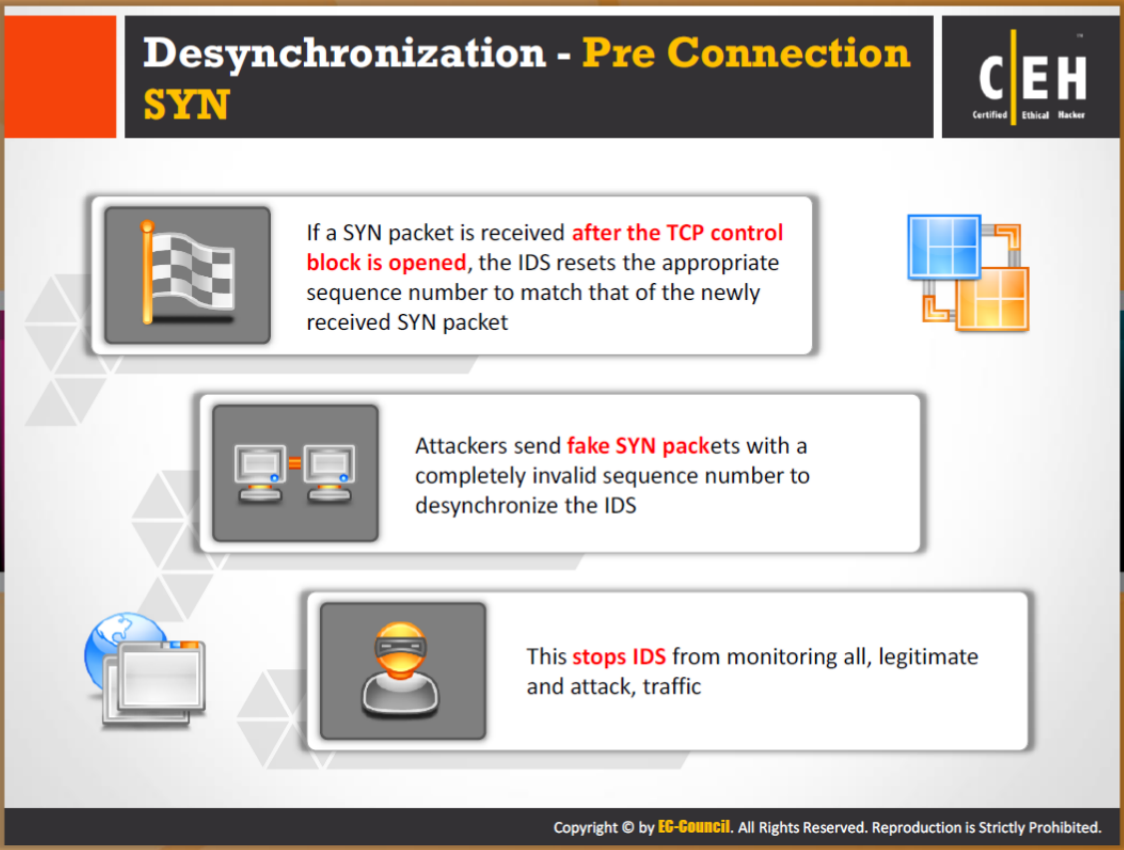
IDS can recognize particular conditions favorable for attack but other alternative forms of attack are also possible, for example, various integer values can be used to exploit integer overflow vulnerabilities

This makes the detection of attack traffic extremely difficult at the IDS

Application-layer Attacks

In order to **transfer media files** speedily, such as images, audios, videos, the files can be compressed and transferred in smaller parts. Attackers find flaws in this **compressed data** and perform attacks and even IDSes cannot identify the signatures within the compressed data.

Many applications that deal with media such as images, video, and, audio employ some form of compression to be sent in a form much smaller than the original, which **increases data transfer speeds**. When a flaw is found in these applications, the entire attack can occur within compressed data, and the IDS can have no way to check the **compressed file format** for signatures. Many IDSes look for specific conditions that allow for an attack. However, there are times when the attack can take many different forms. For example, integer overflow vulnerabilities could be exploited using several different integer values. This fact combined with compressed data makes signature detection extremely difficult.



Desynchronization - Pre Connection SYN

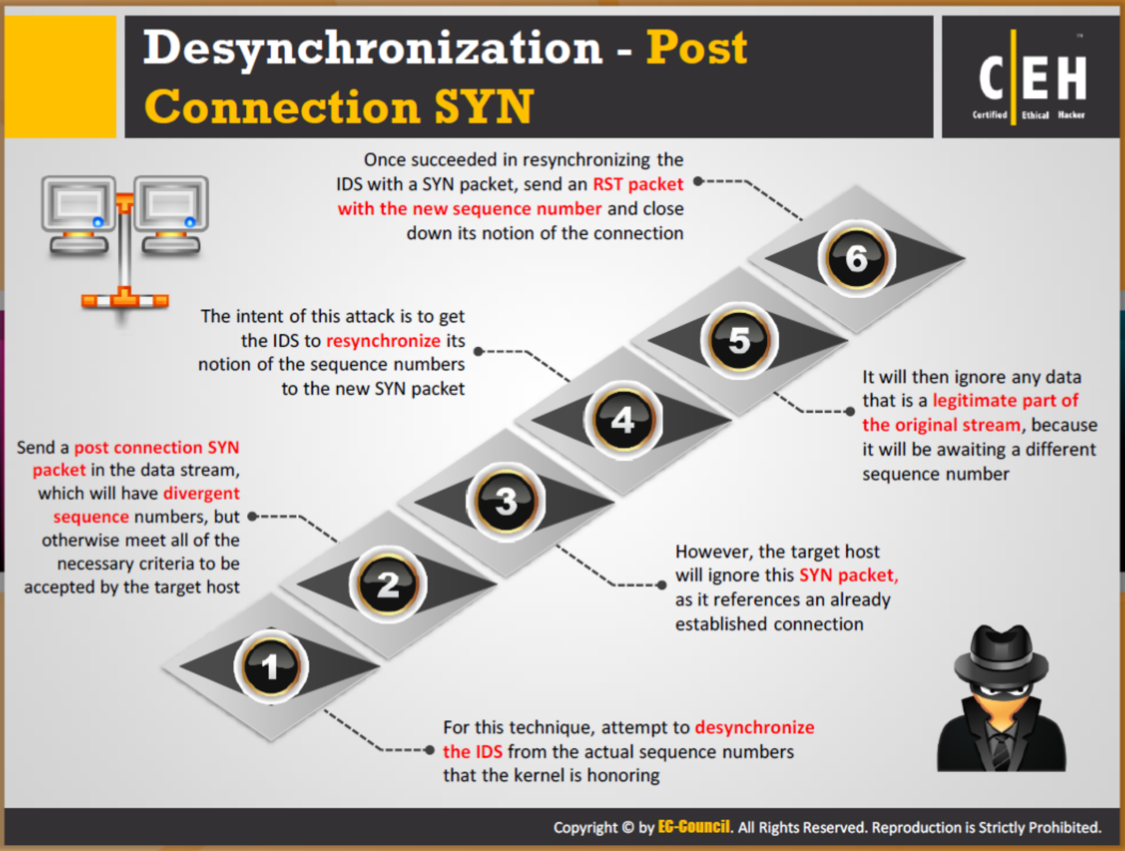
If a SYN packet is received after the TCP control block is opened, the IDS resets the appropriate sequence number to match that of the newly received SYN packet

Attackers send fake SYN packets with a completely invalid sequence number to desynchronize the IDS

This stops IDS from monitoring all, legitimate and attack, traffic

Desynchronization – Pre Connection SYN

This attack calls bind to get the kernel to assign a local port to the socket before calling connect. This is another attack that an attacker performs and sends an initial SYN before the real connection is established, but with an invalid TCP checksum. The sniffer can ignore or accept subsequent SYNs in a connection. If the sniffer is smart, it does not check the TCP checksum; otherwise it checks the TCP checksum. If the sniffer checks the checksum, then the attack is synchronized and a bogus sequence number is sent to the sniffer/IDS before the real connection occurs.



Desynchronization - Post Connection SYN

For this technique, attempt to **desynchronize the IDS** from the actual sequence numbers that the kernel is honoring

Send a **post connection SYN packet** in the data stream, which will have **divergent sequence** numbers, but otherwise meet all of the necessary criteria to be accepted by the target host

However, the target host will ignore this **SYN packet,** as it references an already established connection

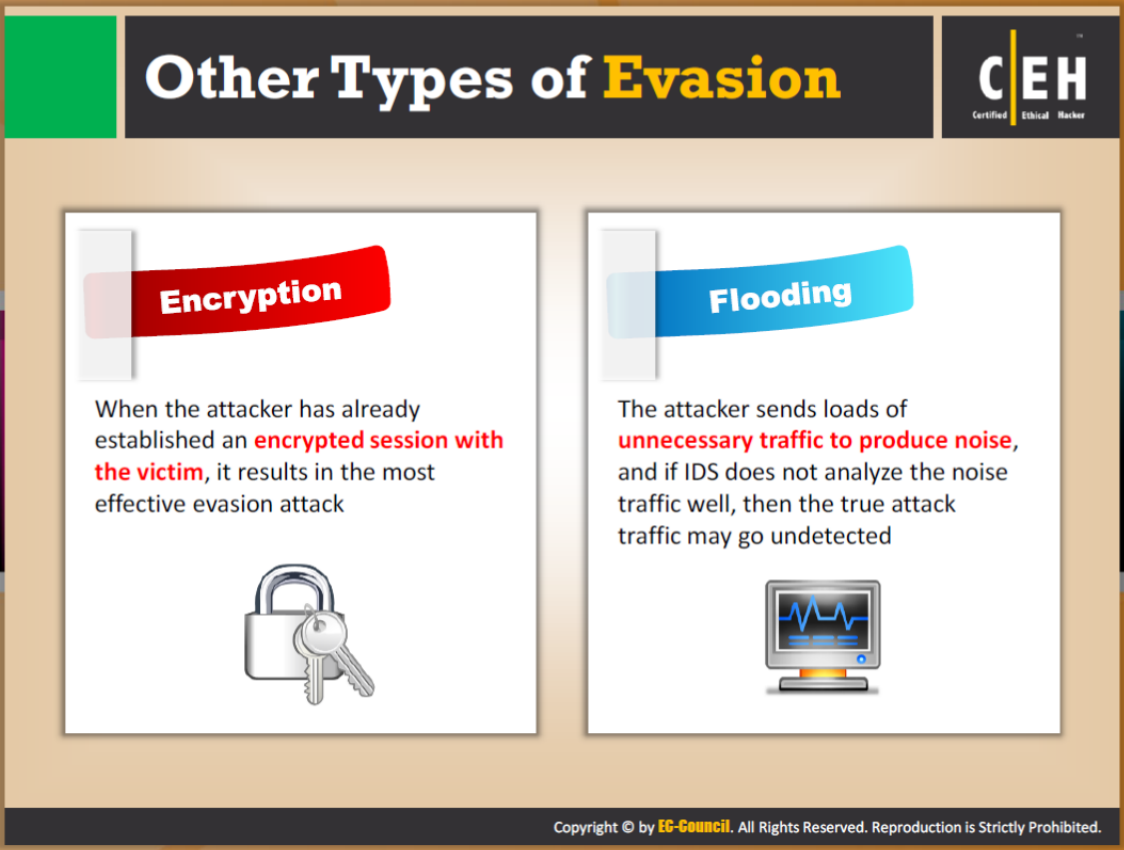
The intent of this attack is to get the IDS to **resynchronize** its notion of the sequence numbers to the new SYN packet

It will then ignore any data that is a **legitimate part of the original stream**, because it will be awaiting a different sequence number

Once succeeded in resynchronizing the IDS with a SYN packet, send an **RST packet with the new sequence number** and close down its notion of the connection

Desynchronization - Post Connection SYN

To deceive an intelligent sniffer or an ID system, attackers do not directly try to deceive it, for it keeps track of the **TCP sequence numbers**. For this technique to work efficiently, attackers first desynchronize the sniffer or IDS. The attack on the sniffer or IDS can be implemented by sending a post connection **SYN packet** in the data stream. The data stream can have all the necessary sequence numbers (all different) and meet the criteria so that the stream is accepted by the target. After transmitting the data stream, the host ignores the SYN packet, because the reference of the SYN packet has already established connection. The motive behind this attack is to resynchronize the **sniffer/IDS**. If the attacker succeeds in resynchronizing the IDS with a SYN packet, attacker then sends an RST packet with the new sequence number.



Other Types of **Evasion**

Encryption

When the attacker has already established an **encrypted session with the victim**, it results in the most effective evasion attack

Flooding

The attacker sends loads of **unnecessary traffic to produce noise**, and if IDS does not analyze the noise traffic well, then the true attack traffic may go undetected

Other Types of Evasion

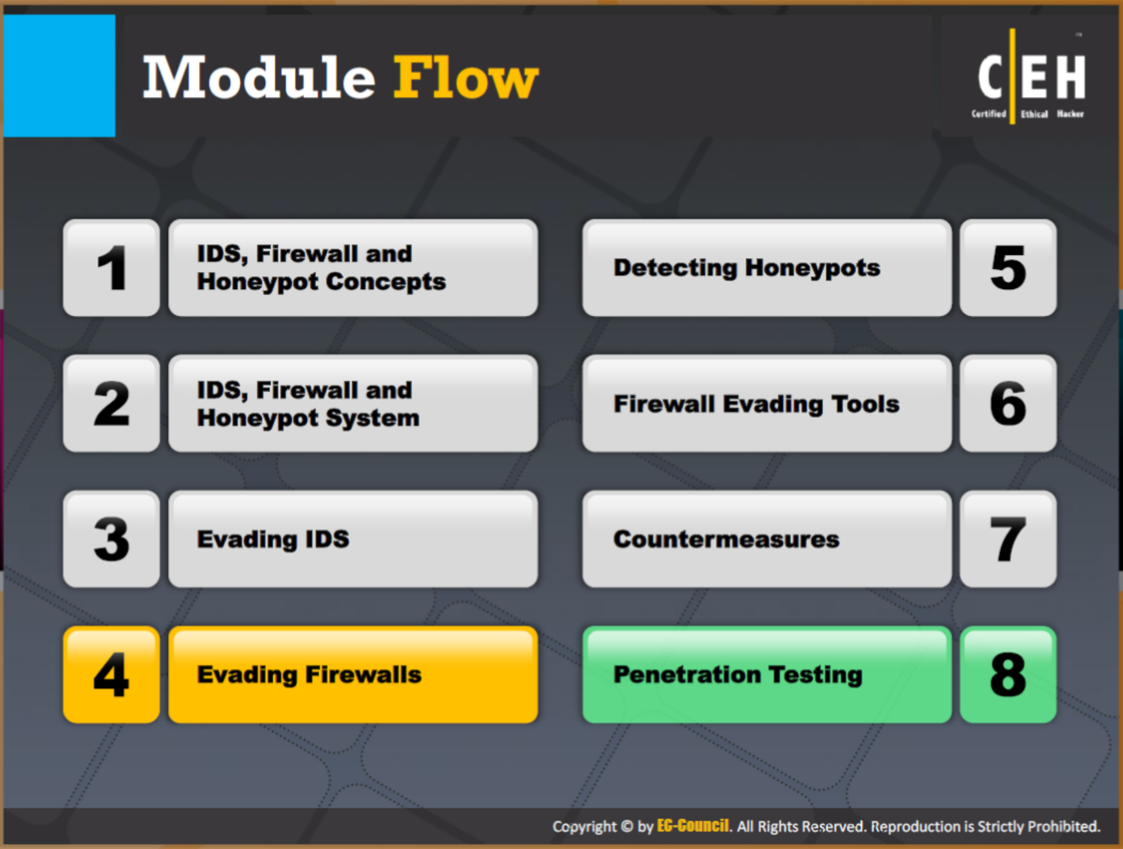
There are two more types of evasion:

Encryption

When the attacker has already established an encrypted session with the victim, it results in the most effective evasion attack.

Flooding

The attacker sends loads of unnecessary traffic to produce noise, and if the IDS do not analyze the noise traffic, the true attack traffic may go undetected.



Module Flow

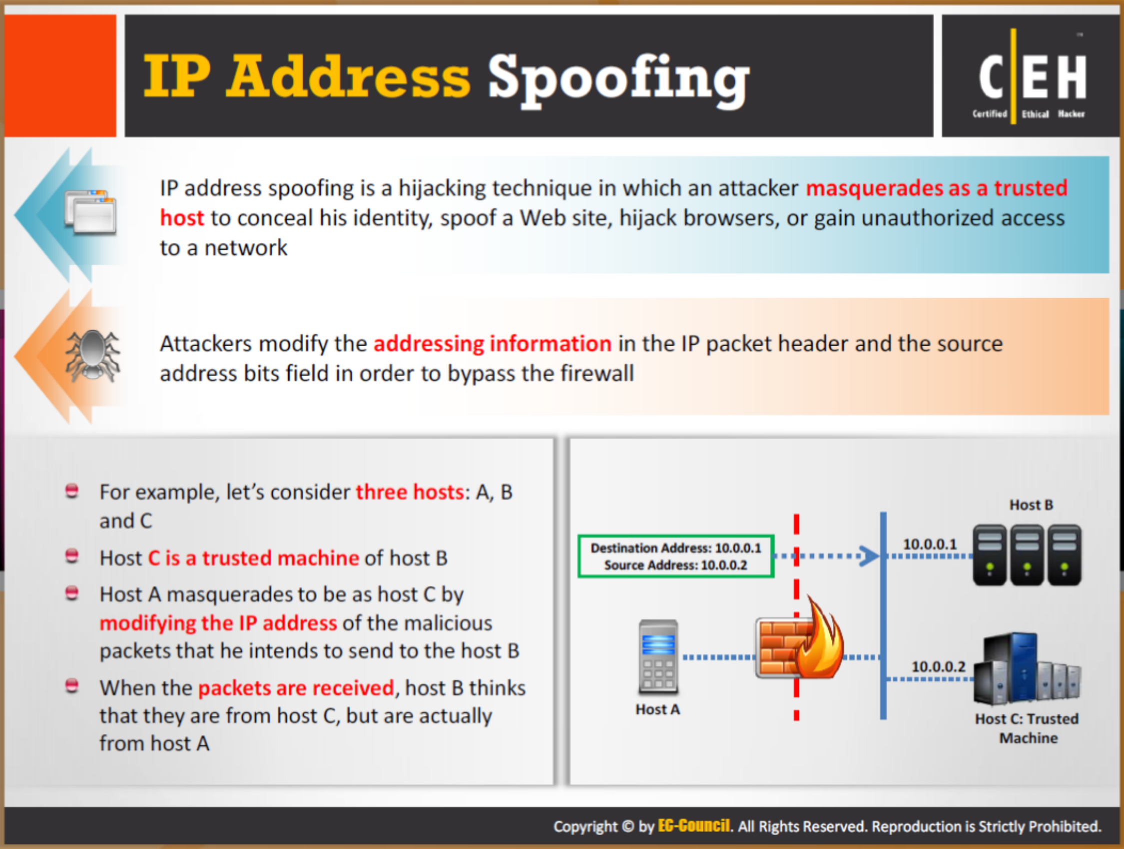
1 IDS, Firewall and Honeypot Concepts
2 IDS, Firewall and Honeypot System
3 Evading IDS
4 Evading Firewalls
5 Detecting Honeypots
6 Firewall Evading Tools
7 Countermeasures
8 Penetration Testing

Module Flow

Firewalls are the security mechanisms implemented by a network or a system to protect itself from being attacked. Attackers try to bypass firewalls so that they can break the security mechanisms and gain access to the legitimate system or network.

IDS, Firewall and Honeypot Concepts	Detecting Honeypots
IDS, Firewall and Honeypot System	Firewall Evading Tools
Evading IDS	Countermeasure
Evading Firewall	Penetration Testing

This section describes various ways in which an attacker can evade the firewall.



IP Address Spoofing

IP address spoofing is a hijacking technique in which an attacker **masquerades as a trusted host** to conceal his identity, spoof a Web site, hijack browsers, or gain unauthorized access to a network

Attackers modify the **addressing information** in the IP packet header and the source address bits field in order to bypass the firewall

- For example, let's consider **three hosts**: A, B and C
- Host **C is a trusted machine** of host B
- Host A masquerades to be as host C by **modifying the IP address** of the malicious packets that he intends to send to the host B
- When the **packets are received**, host B thinks that they are from host C, but are actually from host A

IP Address Spoofing

IP address spoofing or IP spoofing is one of the ways that an attacker tries to evade **firewall restrictions**. IP spoofing is a technique where the **attacker creates Internet protocol** packets by using a forged IP address and gains access over the system or network without any **authorization**. The attacker spoofs the messages and they appear to be sent from a reliable source. Thus, the attacker succeeds in impersonating others' identities with help of IP spoofing. Hackers generally use this technique for not getting caught while spamming and various other activities.

The following scenario shows how an attacker bypasses a firewall by impersonating a different identity with the help of th IP spoofing technique:

- Let's consider three hosts: A, B, and C
- Host C is a trusted machine of host B
- Host A wants to send some packets to host B and A impersonates itself to be C by changing the IP address of these packets
- When these packets are received, B thinks that these packets are from C, but actually they are from A

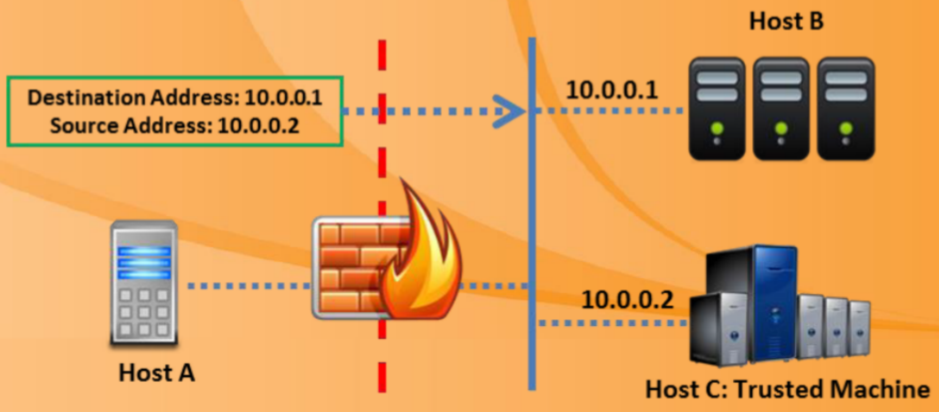
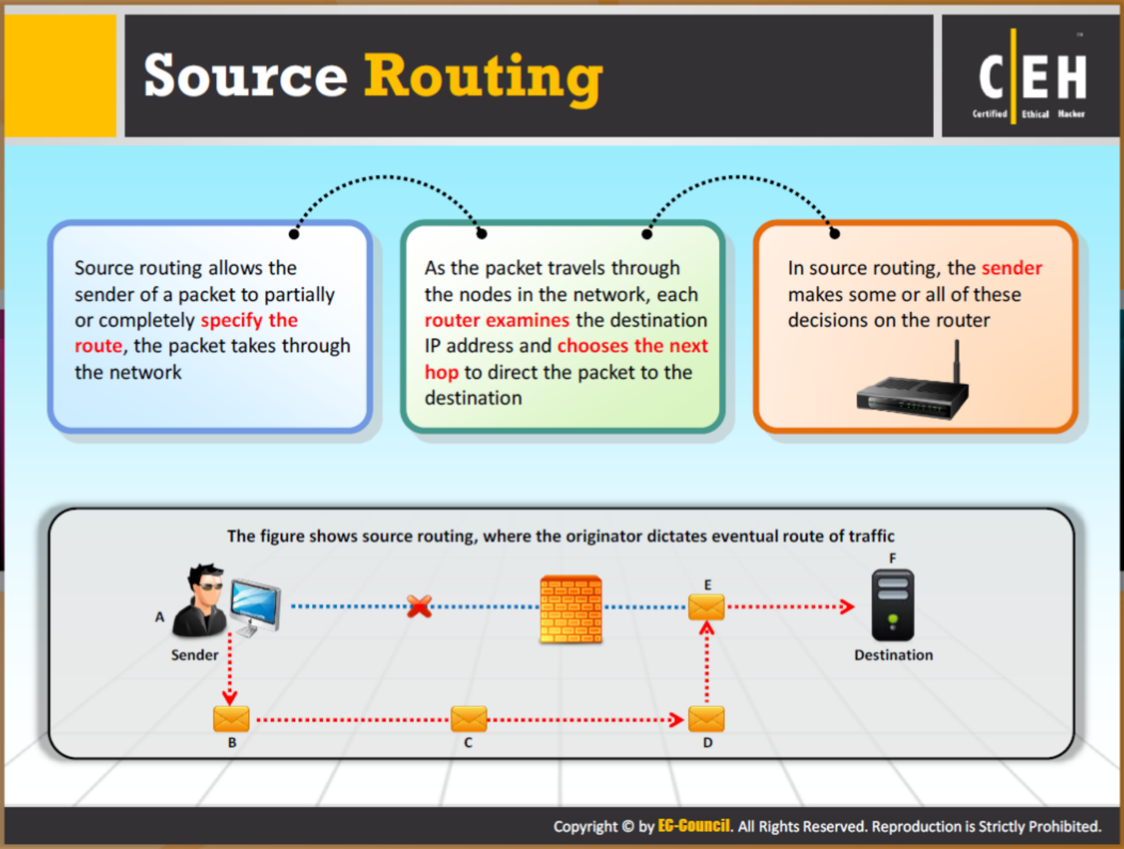

Host B
Destination Address: 10.0.0.1
Source Address: 10.0.0.2
10.0.0.1
Host A
10.0.0.2
Host C: Trusted Machine

FIGURE 17.27: Working of IP Address Spoofing



Source Routing

Source routing allows the sender of a packet to partially or completely specify the route, the packet takes through the network

As the packet travels through the nodes in the network, each router examines the destination IP address and chooses the next hop to direct the packet to the destination

In source routing, the sender makes some or all of these decisions on the router

The figure shows source routing, where the originator dictates eventual route of traffic

Source Routing

Using this technique, the sender of the packet designates the route that a packet should take through the network in such a way that the designated route should bypass the firewall node. Using this technique the attacker can evade the firewall restrictions.

When these packets travel through the nodes in the network, each router will check the IP address of the destination and choose the next node to forward them. In source routing, the sender makes some or all of these decisions on the router.

The figure shows the principle of the source routing but it is an optimal way, which makes the decision of the next hop.

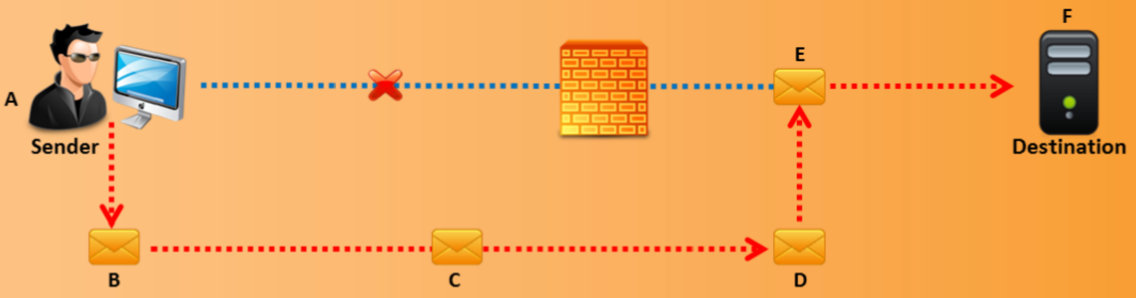
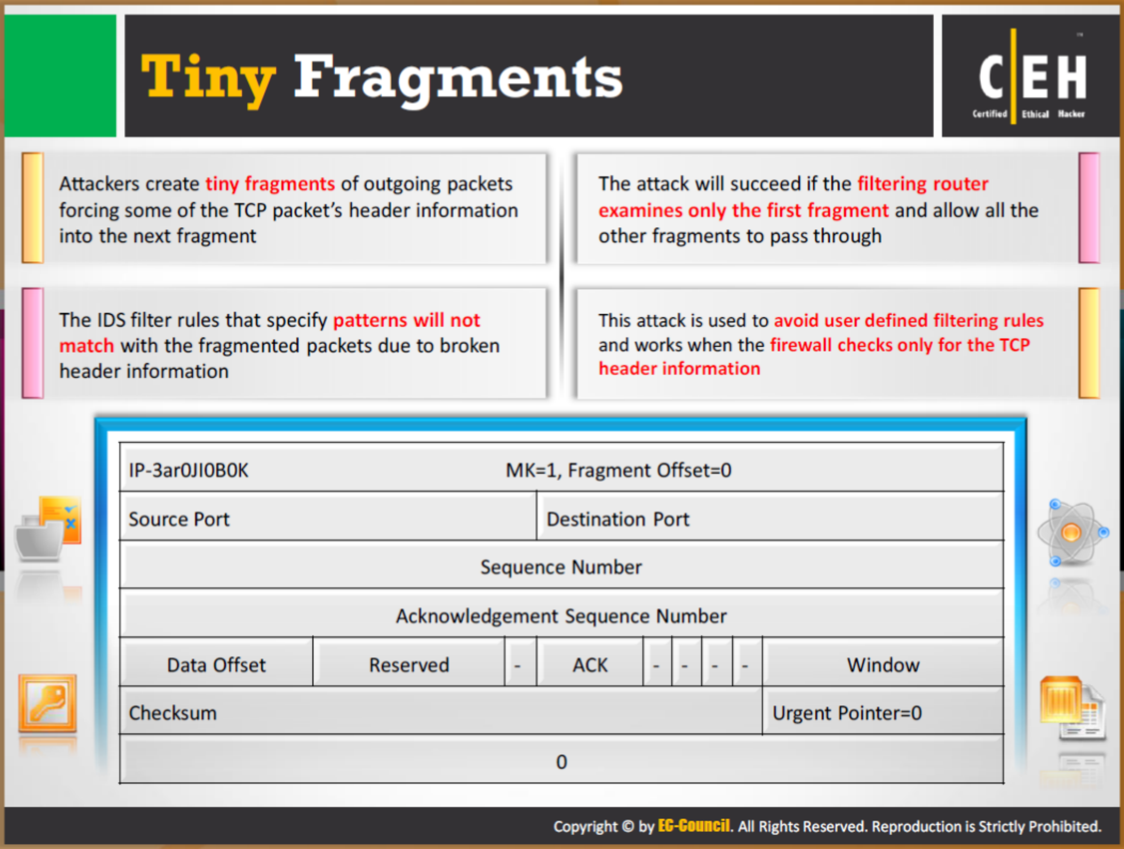


FIGURE 17.28:



Tiny Fragments

Attackers create tiny fragments of outgoing packets forcing some of the TCP packet's header information into the next fragment

The attack will succeed if the filtering router examines only the first fragment and allow all the other fragments to pass through

The IDS filter rules that specify patterns will not match with the fragmented packets due to broken header information

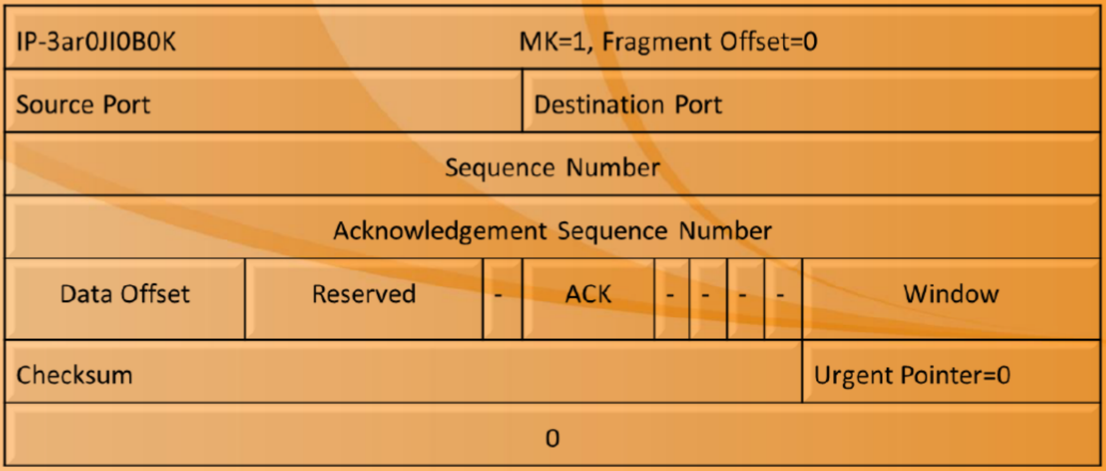
This attack is used to avoid user defined filtering rules and works when the firewall checks only for the TCP header information

IP-3ar0JIOBOK	MK=1, Fragment Offset=0							
Source Port			Destination Port					
Sequence Number								
Acknowledgement Sequence Number								
Data Offset	Reserved	-	ACK	-	-	-	-	Window
Checksum								Urgent Pointer=0
0								

Tiny Fragments

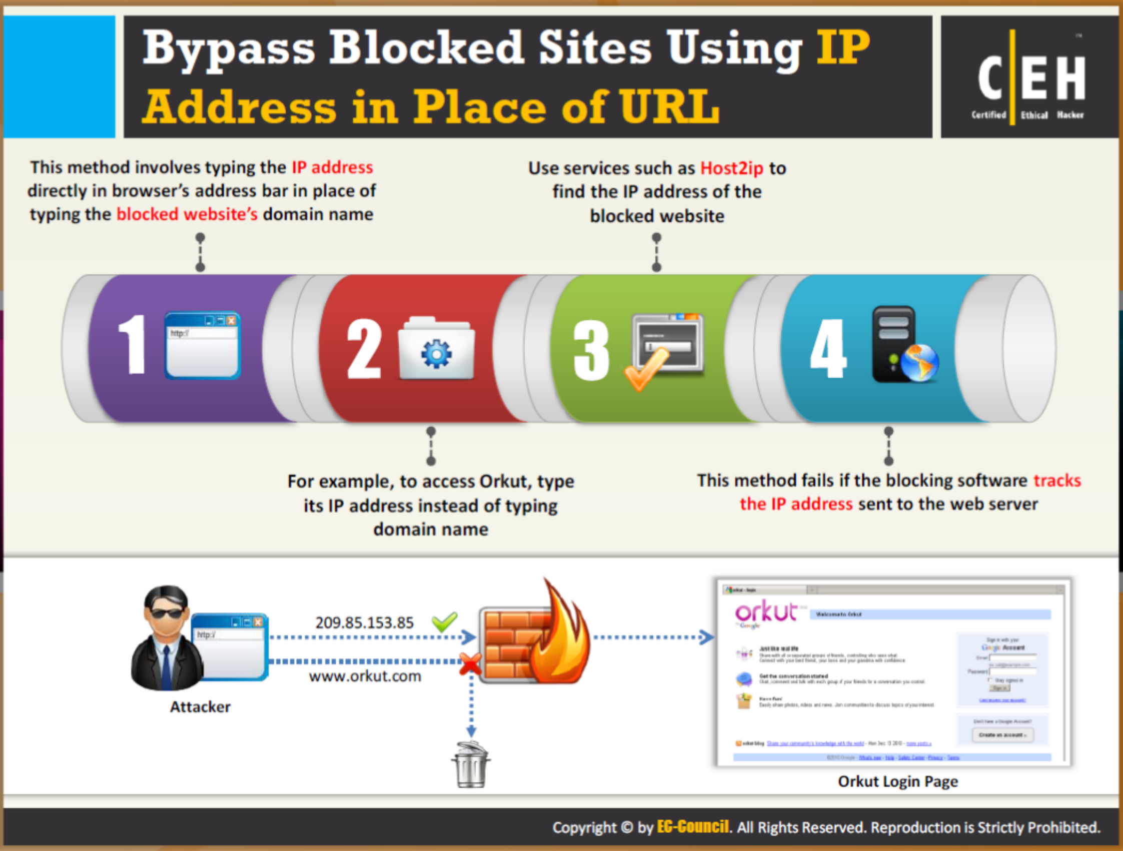
The attacker uses the **IP fragmentation technique** to create extremely small fragments and force the **TCP header** information into the next fragment. This may result in a case whereby the TCP flags field is forced into the second fragment, and filters will be unable to check these flags in the first octet thus ignoring them in **subsequent fragments**.

Attackers hope that only the first fragment is examined by the **filtering router (firewall)** and the remaining fragments are passed through. This attack is used to avoid user defined filtering rules and works when the firewall checks only for the TCP header information.



IP-3ar0JI0B0K			MK=1, Fragment Offset=0					
Source Port			Destination Port					
Sequence Number								
Acknowledgement Sequence Number								
Data Offset	Reserved	-	ACK	-	-	-	-	Window
Checksum								Urgent Pointer=0
0								

FIGURE 17.29: Tiny Fragments Diagram



Bypass Blocked Sites Using IP Address in Place of URL

This method involves typing the IP address directly in browser's address bar in place of typing the blocked website's domain name

Use services such as Host2ip to find the IP address of the blocked website

1 2 3 4

For example, to access Orkut, type its IP address instead of typing domain name

This method fails if the blocking software tracks the IP address sent to the web server

Bypass Blocked Sites Using IP Address in Place of URL

You can also evade firewall restrictions by typing the IP address of the blocked site instead of its domain names. This allows you to access the restricted or blocked sites. You need to use some tools to convert the target domain name into its IP address.

For example:

- Instead of typing www.Orkut.com, type its IP address to access Orkut
- Host2ip can help you to find the IP address of that blocked website
- If the blocking software can track the IP address sent to the web server, the website could not be unblocked or accessed by using this method

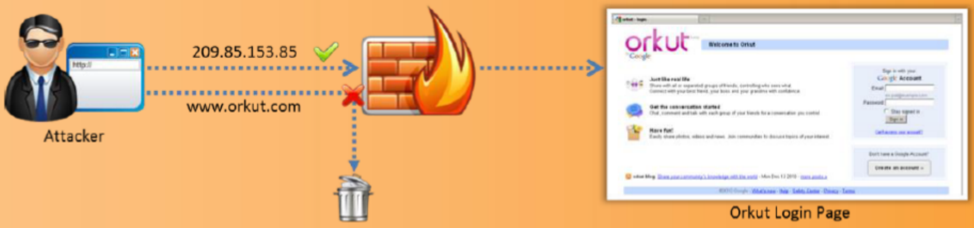
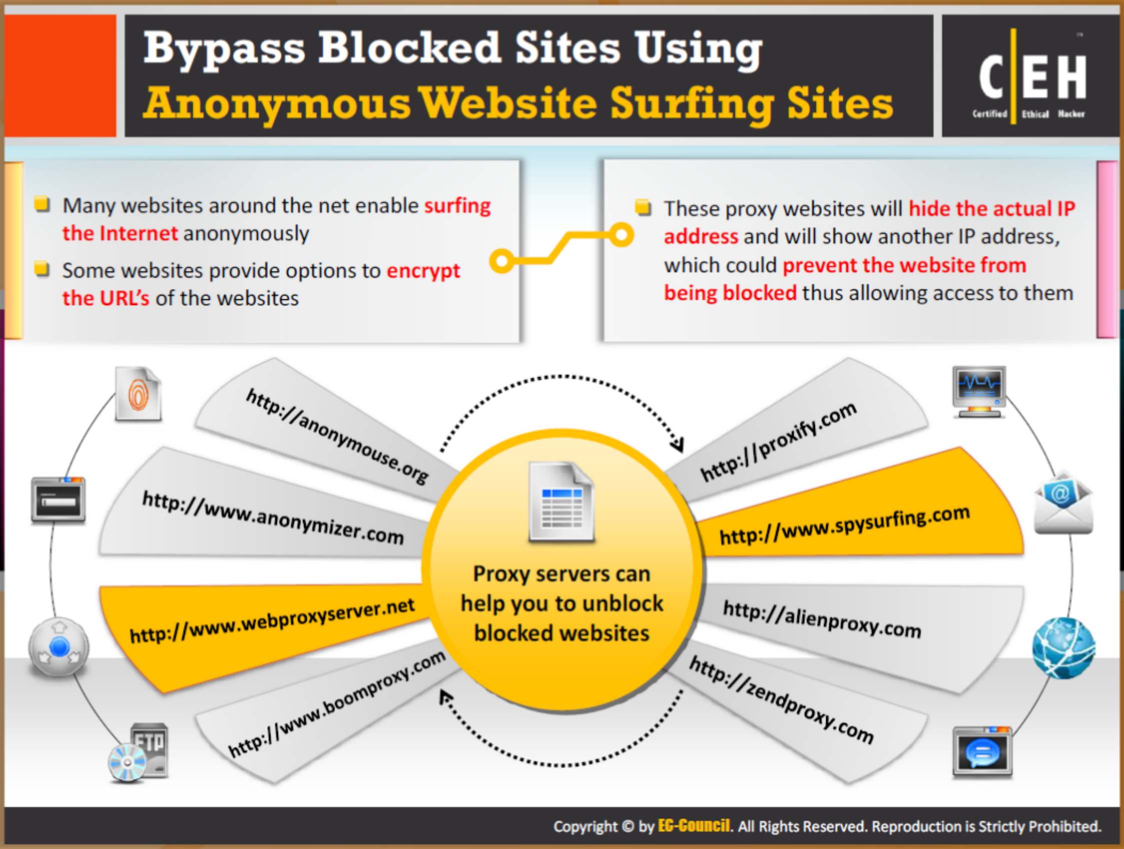


FIGURE 17.30: Bypass Blocked Sites Using IP Address in Place of URL



Bypass Blocked Sites Using Anonymous Website Surfing Sites

- Many websites around the net enable surfing the Internet anonymously
- Some websites provide options to encrypt the URL's of the websites

- These proxy websites will hide the actual IP address and will show another IP address, which could prevent the website from being blocked thus allowing access to them

Proxy servers can help you to unblock blocked websites

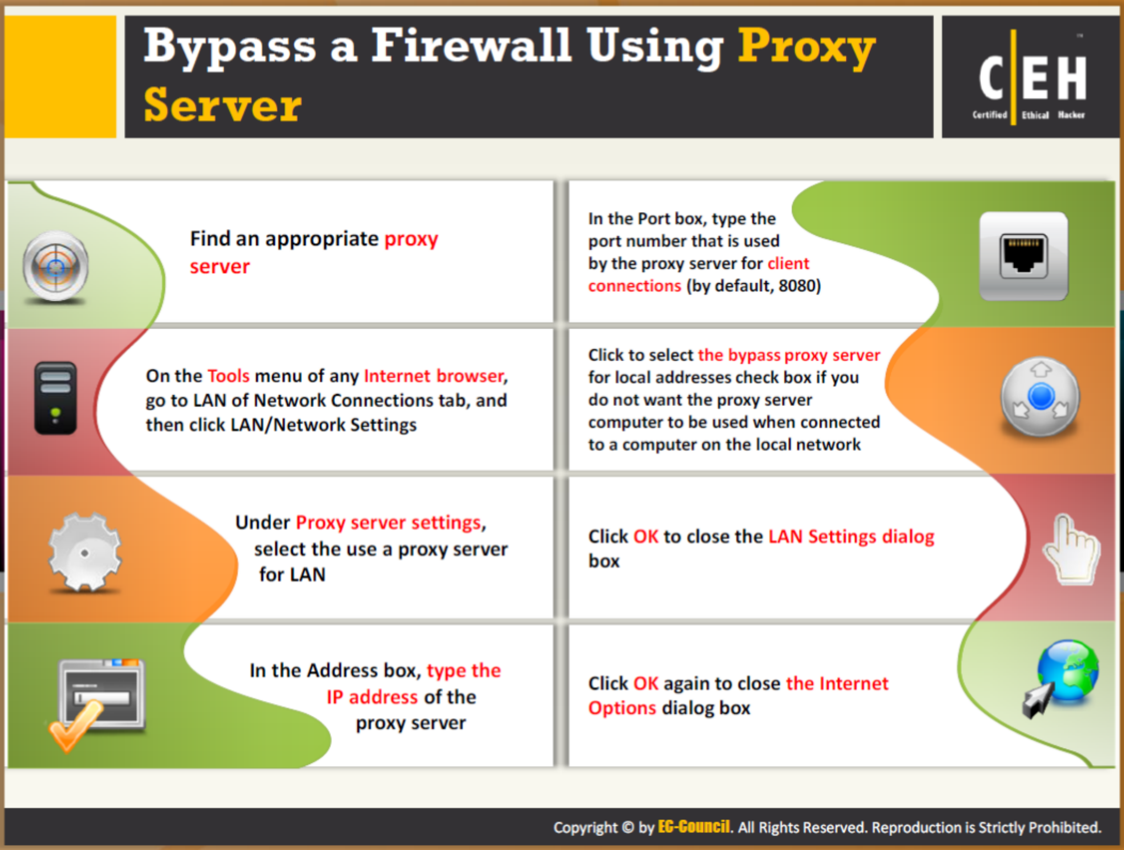
http://anonymouse.org
http://www.anonymizer.com
http://www.webproxyserver.net
http://www.boomproxy.com
http://proxify.com
http://www.spysurfing.com
http://alienproxy.com
http://zendproxy.com

Bypass Blocked Sites Using Anonymous Website Surfing Sites

Anonymous website surfing sites help you to surf the Internet anonymously and to unblock blocked sites. i.e., evade **firewall restrictions**. By using these sites, you can surf restricted sites anonymously, i.e., without using your IP address on the Internet. There are a number of anonymous **website surfing sites** available on the Internet. Some websites provide options to encrypt the URLs of websites.

Here is a list of some of the proxy servers that can help you to unblock blocked websites:

- http://anonymouse.org
- http://www.anonymizer.com
- http://www.webproxyserver.net
- http://www.boomproxy.com
- http://proxify.com
- http://www.spysurfing.com
- http://alienproxy.com
- http://zendproxy.com



Bypass a Firewall Using Proxy Server

Find an appropriate proxy server

On the Tools menu of any Internet browser, go to LAN of Network Connections tab, and then click LAN/Network Settings

Under Proxy server settings, select the use a proxy server for LAN

In the Address box, type the IP address of the proxy server

In the Port box, type the port number that is used by the proxy server for client connections (by default, 8080)

Click to select the bypass proxy server for local addresses check box if you do not want the proxy server computer to be used when connected to a computer on the local network

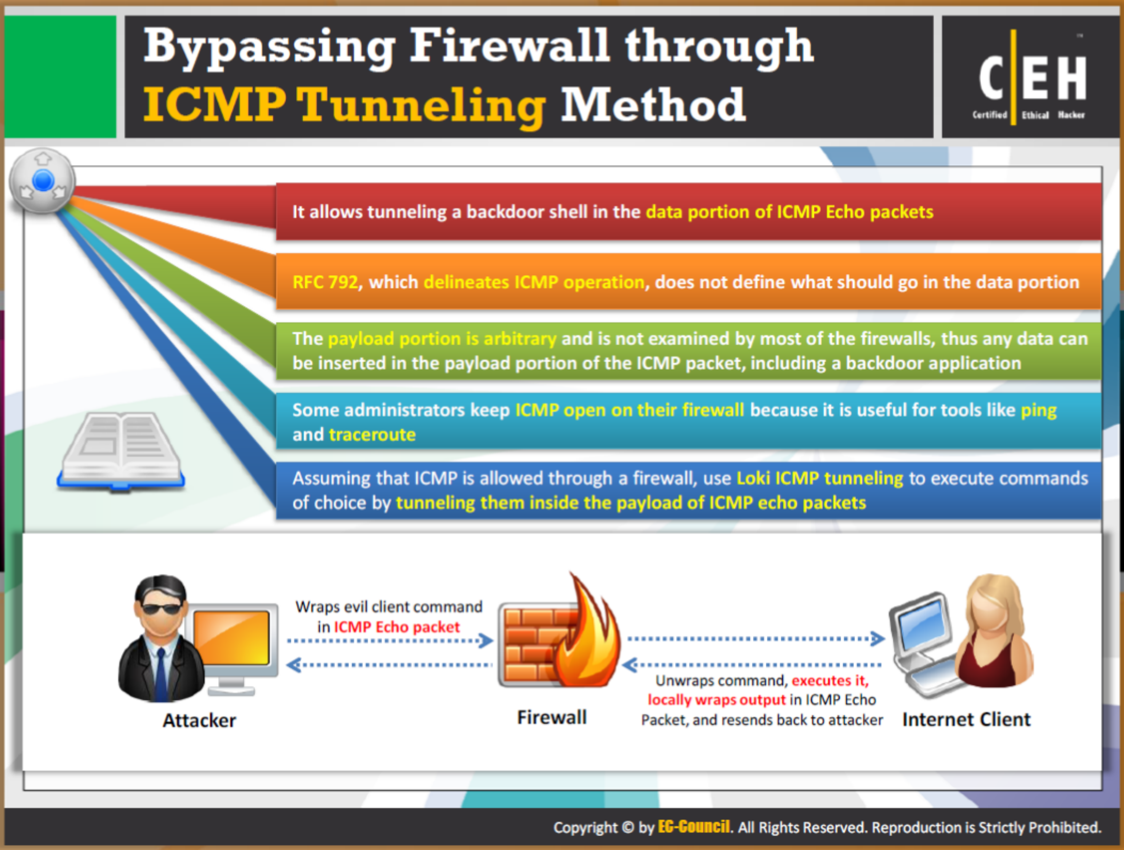
Click OK to close the LAN Settings dialog box

Click OK again to close the Internet Options dialog box

Bypass a Firewall Using a Proxy Server

By using a proxy server, you can also bypass the firewall restriction imposed by a particular organization. To evade the firewall restrictions using a proxy server, follow these steps:

1. Find an appropriate proxy server.
2. On the **Tools** menu of any Internet browser, go to LAN of Network Connections tab, and then click **LAN/Network Settings.**
3. Under **Proxy server settings**, select the use a proxy server for the LAN.
4. In the **Address** text box, type the IP address of the proxy server.
5. In the **Port** text box, type the port number that is used by the proxy server for client connections (by default, 8080).
6. Click to select the bypass proxy server for local addresses check box if you do not want the proxy server computer to be used when connected to a computer on the local network.
7. Click **OK** to close the **LAN Settings** dialog box.
8. Click **OK** again to close the **Internet Options** dialog box.



Bypassing Firewall through ICMP Tunneling Method

It allows tunneling a backdoor shell in the data portion of ICMP Echo packets

RFC 792, which delineates ICMP operation, does not define what should go in the data portion

The payload portion is arbitrary and is not examined by most of the firewalls, thus any data can be inserted in the payload portion of the ICMP packet, including a backdoor application

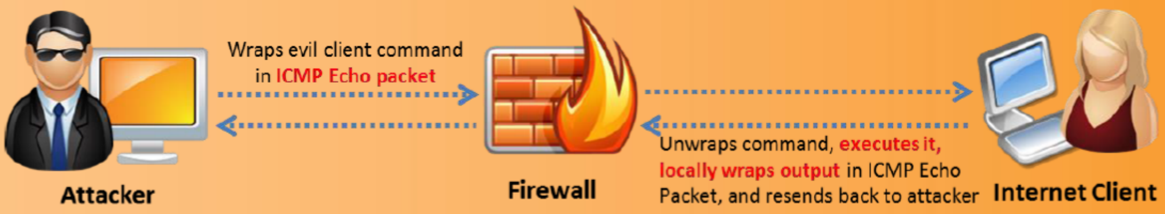
Some administrators keep ICMP open on their firewall because it is useful for tools like ping and traceroute

Assuming that ICMP is allowed through a firewall, use Loki ICMP tunneling to execute commands of choice by tunneling them inside the payload of ICMP echo packets

C|EH
Certified Ethical Hacker
Wraps evil client command in ICMP Echo packet
Attacker
Firewall
Unwraps command, executes it, locally wraps output in ICMP Echo Packet, and resends back to attacker
Internet Client

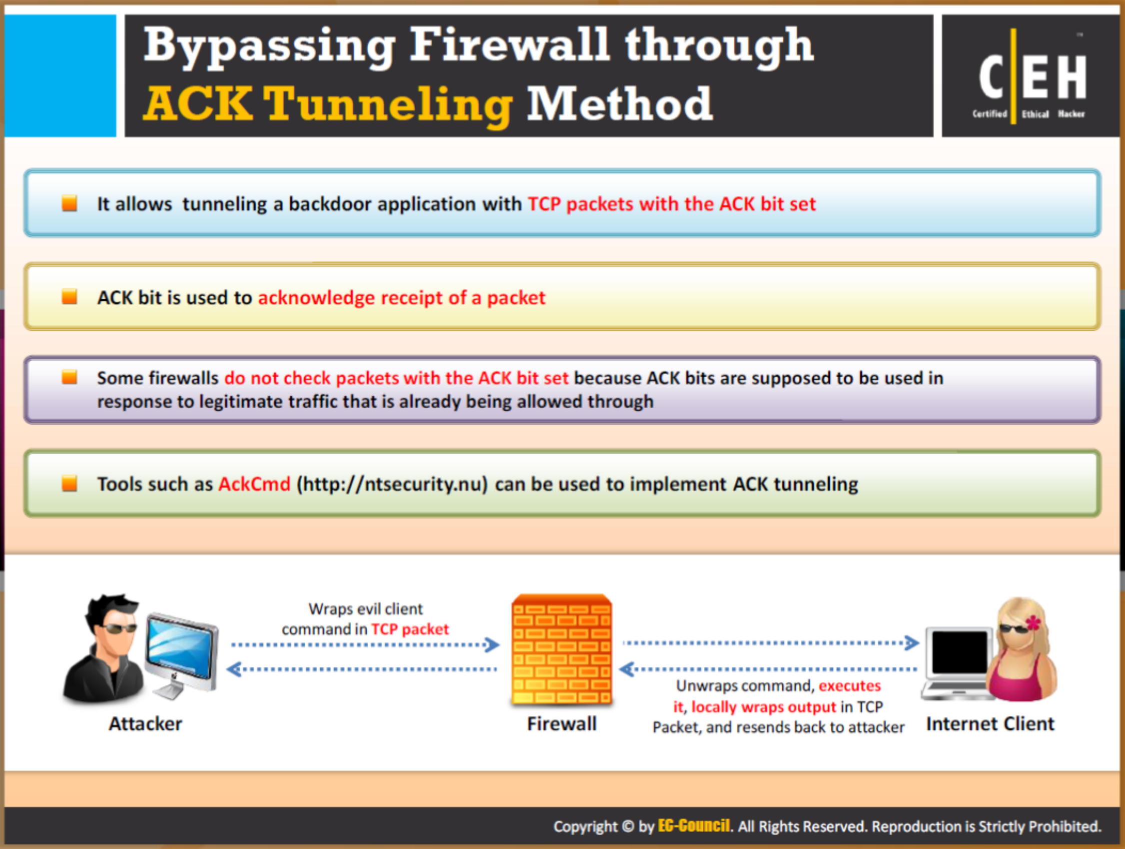
Bypassing a Firewall through the ICMP Tunneling Method

ICMP tunneling allows tunneling a **backdoor shell** in the data portion of ICMP Echo packets. **RFC 792**, which delineates ICMP operation, does not define what should go in the data portion. The payload portion is arbitrary and is not examined by most of the firewalls, thus any data can be inserted in the payload portion of the ICMP packet, including a **backdoor application**. Some administrators keep ICMP open on their firewall because it is useful for tools like ping and traceroute. Assuming that ICMP is allowed through a firewall, use **Loki ICMP tunneling** to execute commands of choice by tunneling them inside the payload of ICMP echo packets



Wraps evil client command in ICMP Echo packet
Attacker
Firewall
Unwraps command, executes it, locally wraps output in ICMP Echo Packet, and resends back to attacker
Internet Client

FIGURE 17.31: Bypassing a Firewall through the ICMP Tunneling Method



Bypassing Firewall through ACK Tunneling Method

- It allows tunneling a backdoor application with TCP packets with the ACK bit set
- ACK bit is used to acknowledge receipt of a packet
- Some firewalls do not check packets with the ACK bit set because ACK bits are supposed to be used in response to legitimate traffic that is already being allowed through
- Tools such as AckCmd (http://ntsecurity.nu) can be used to implement ACK tunneling

Wraps evil client command in TCP packet

Attacker

Firewall

Unwraps command, executes it, locally wraps output in TCP Packet, and resends back to attacker

Internet Client

Bypassing a Firewall through the ACK Tunneling Method

ACK tunneling allows tunneling a backdoor application with TCP packets with the ACK bit set. The ACK bit is used to acknowledge receipt of a packet. Some **firewalls** do not check packets with the ACK bit set because **ACK bits** are supposed to be used in response to legitimate traffic that is already being allowed through. Attackers use this as an advantage to perform ACK tunneling. Tools such as **AckCmd (http://ntsecurity.nu)** can be used to implement ACK tunneling.

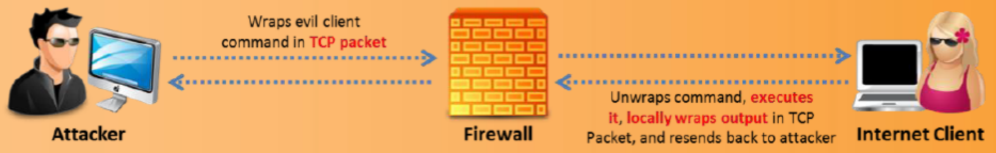
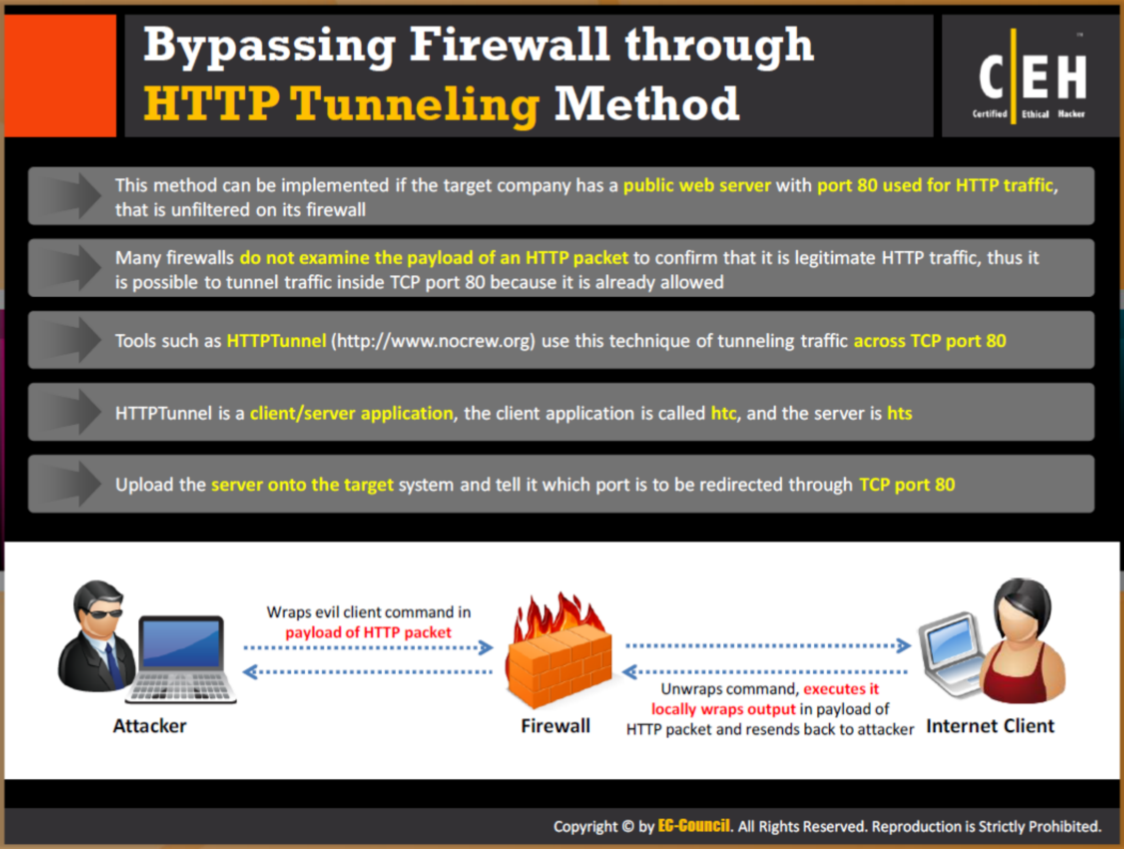


FIGURE 17.32: Bypassing a Firewall through the ACK Tunneling Method



Bypassing Firewall through HTTP Tunneling Method

- This method can be implemented if the target company has a **public web server** with **port 80 used for HTTP traffic**, that is unfiltered on its firewall
- Many firewalls **do not examine the payload of an HTTP packet** to confirm that it is legitimate HTTP traffic, thus it is possible to tunnel traffic inside TCP port 80 because it is already allowed
- Tools such as **HTTPTunnel** (http://www.nocrew.org) use this technique of tunneling traffic **across TCP port 80**
- HTTPTunnel is a **client/server application**, the client application is called **htc**, and the server is **hts**
- Upload the **server onto the target** system and tell it which port is to be redirected through **TCP port 80**

Attacker
Wraps evil client command in payload of HTTP packet
Firewall
Unwraps command, executes it locally wraps output in payload of HTTP packet and resends back to attacker
Internet Client

Bypassing a Firewall through the HTTP Tunneling Method

This method can be implemented if the target company has a public web server with port 80 used for HTTP traffic, that is unfiltered on its firewall. Many firewalls do not examine the payload of an HTTP packet to confirm that it is legitimate HTTP traffic, thus it is possible to tunnel traffic inside **TCP port 80** because it is already allowed.

Tools such as **HTTPTunnel (http://www.nocrew.org)** use this technique of tunneling traffic across TCP port 80. HTTPTunnel is a client/server application, the client application is called htc, and the server is hts. Upload the server onto the target system and tell it which port is to be redirected through TCP port 80.

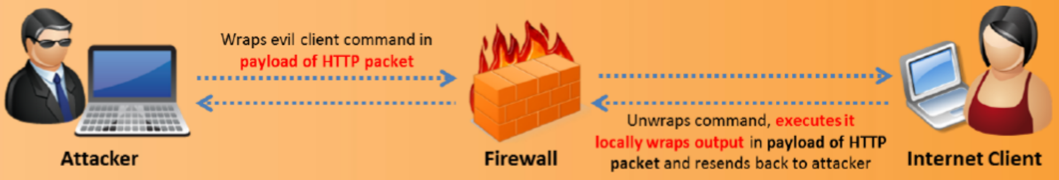
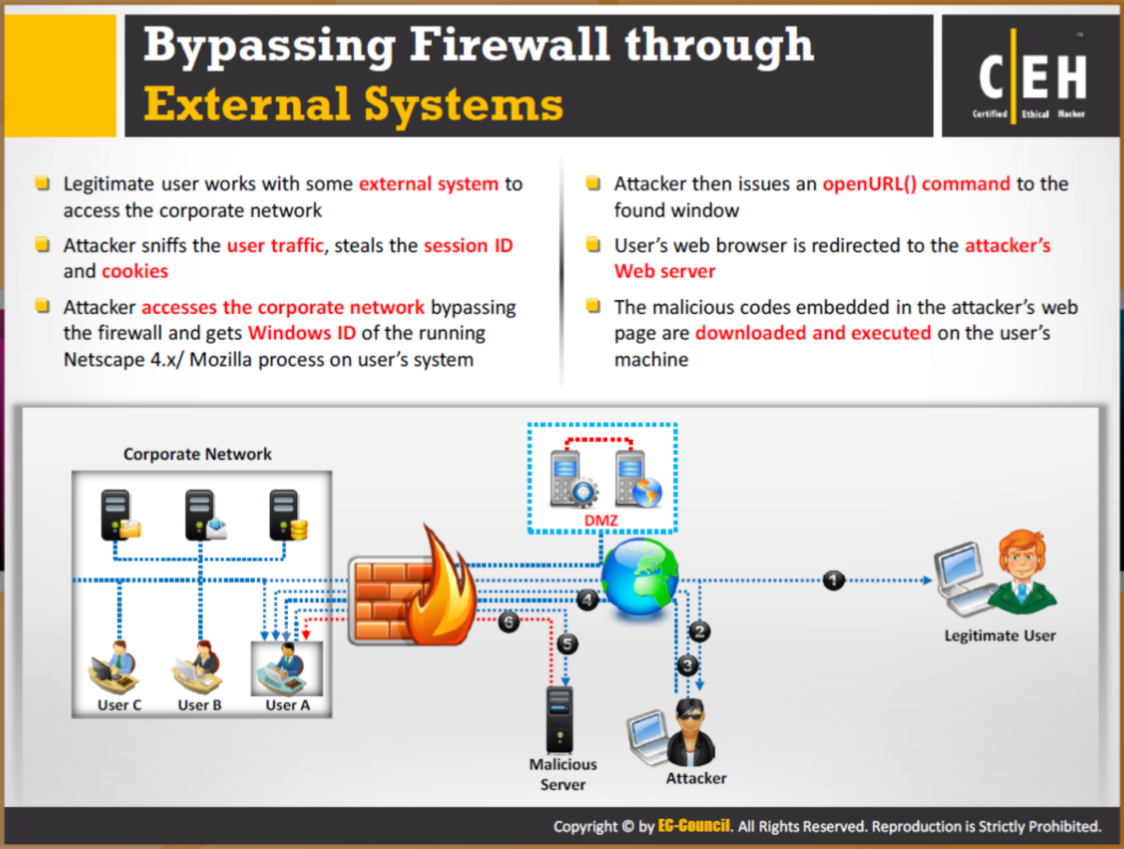

Wraps evil client command in
payload of HTTP packet
Attacker
Firewall
Unwraps command, executes it
locally wraps output in payload of HTTP
packet and resends back to attacker
Internet Client

FIGURE 17.33: Bypassing a Firewall through the HTTP Tunneling Method



Bypassing Firewall through External Systems

- Legitimate user works with some external system to access the corporate network
- Attacker sniffs the user traffic, steals the session ID and cookies
- Attacker accesses the corporate network bypassing the firewall and gets Windows ID of the running Netscape 4.x/ Mozilla process on user's system
- Attacker then issues an openURL() command to the found window
- User's web browser is redirected to the attacker's Web server
- The malicious codes embedded in the attacker's web page are downloaded and executed on the user's machine

Bypassing a Firewall through External Systems

Attackers can bypass firewall restrictions through external systems as follows:

1. Legitimate user works with some external system to access the corporate network.
2. Attacker sniffs the user traffic, and steals the session ID and cookies.
3. Attacker accesses the corporate network bypassing the firewall and gets Windows ID of the running Netscape 4.x/ Mozilla process on user's system.
4. Attacker then issues an openURL() command to the found window.
5. User's web browser connects with the attacker's WWW server.
6. Attacker inserts malicious payload into the requested web page (Java applet) and thus the attacker's code gets executed on the user's machine.

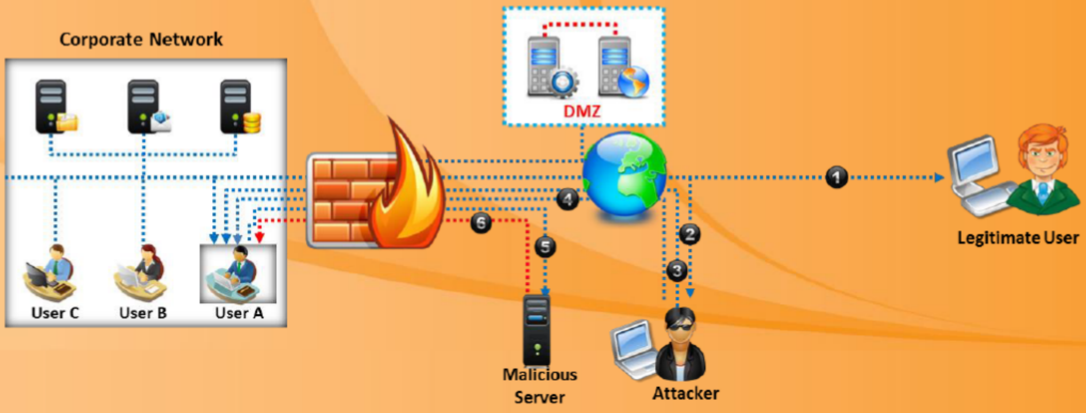
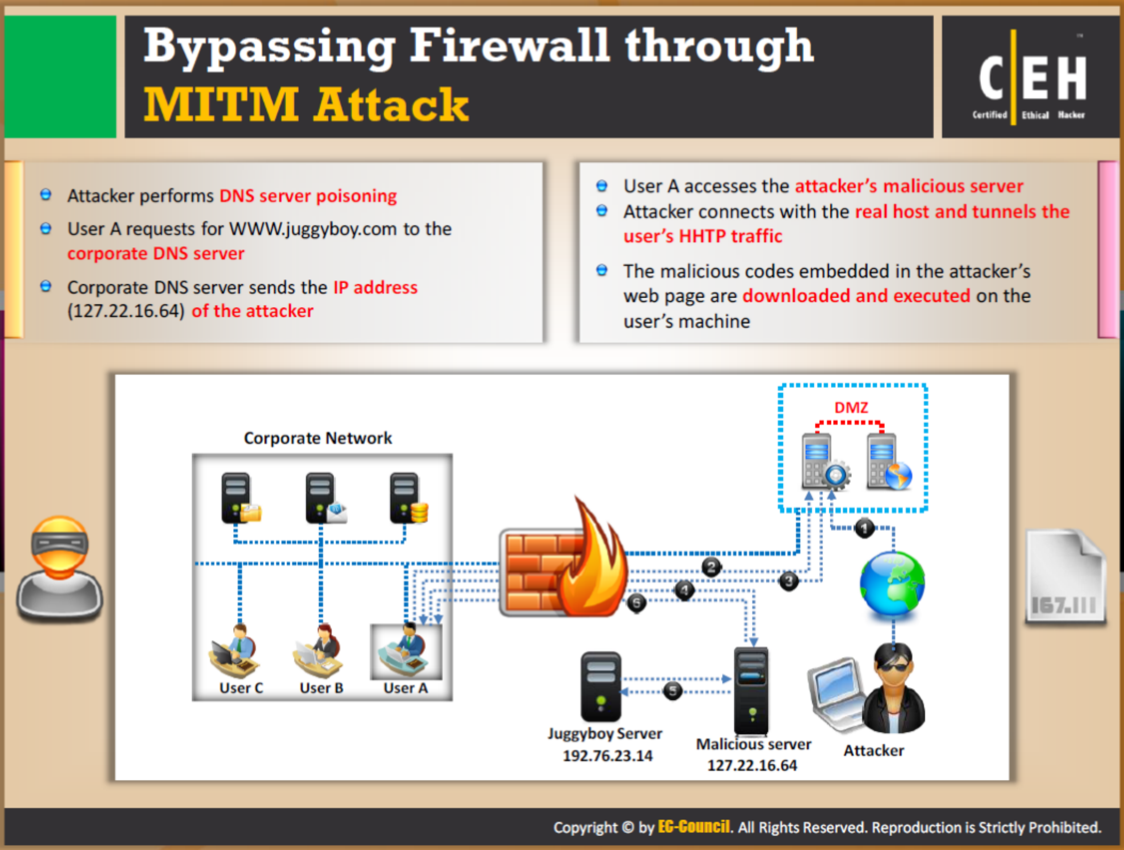

Corporate Network
DMZ
1
2
3
4
5
6
Legitimate User
User C
User B
User A
Malicious Server
Attacker

FIGURE 17.34: Bypassing a Firewall through External Systems



Bypassing Firewall through MITM Attack

- Attacker performs DNS server poisoning
- User A requests for WWW.juggyboy.com to the corporate DNS server
- Corporate DNS server sends the IP address (127.22.16.64) of the attacker
- User A accesses the attacker's malicious server
- Attacker connects with the real host and tunnels the user's HHTP traffic
- The malicious codes embedded in the attacker's web page are downloaded and executed on the user's machine

C|EH
Certified Ethical Hacker
Corporate Network
DMZ
User C
User B
User A
❶
❷
❸
❹
❺
❻
Juggyboy Server
192.76.23.14
Malicious server
127.22.16.64
Attacker

Bypassing a Firewall through a MITM Attack

The following steps illustrate an example scenario of how an attacker bypasses a firewall through an MITM attack:

1. Attacker performs DNS server poisoning.
2. User A requests WWW.juggyboy.com to the corporate DNS server.
3. Corporate DNS server sends the IP address (127.22.16.64) of the attacker.
4. User A accesses the attacker's malicious server.
5. Attacker connects with the real host and tunnels the user's HHTP traffic.
6. Attacker inserts malicious payload into the requested web page (Java applet), and thus the attacker's code is executed on the user's machine.

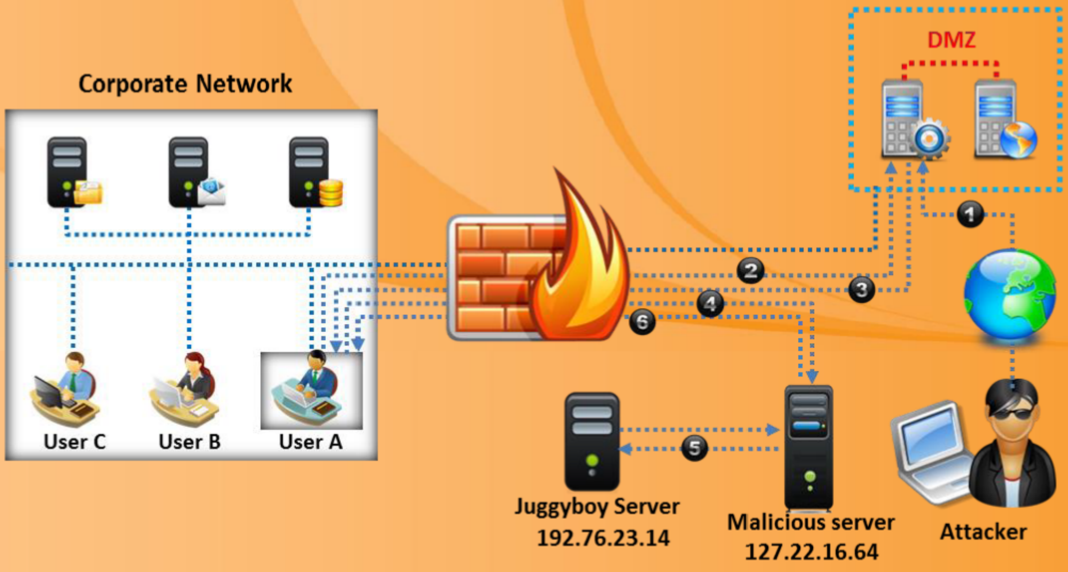
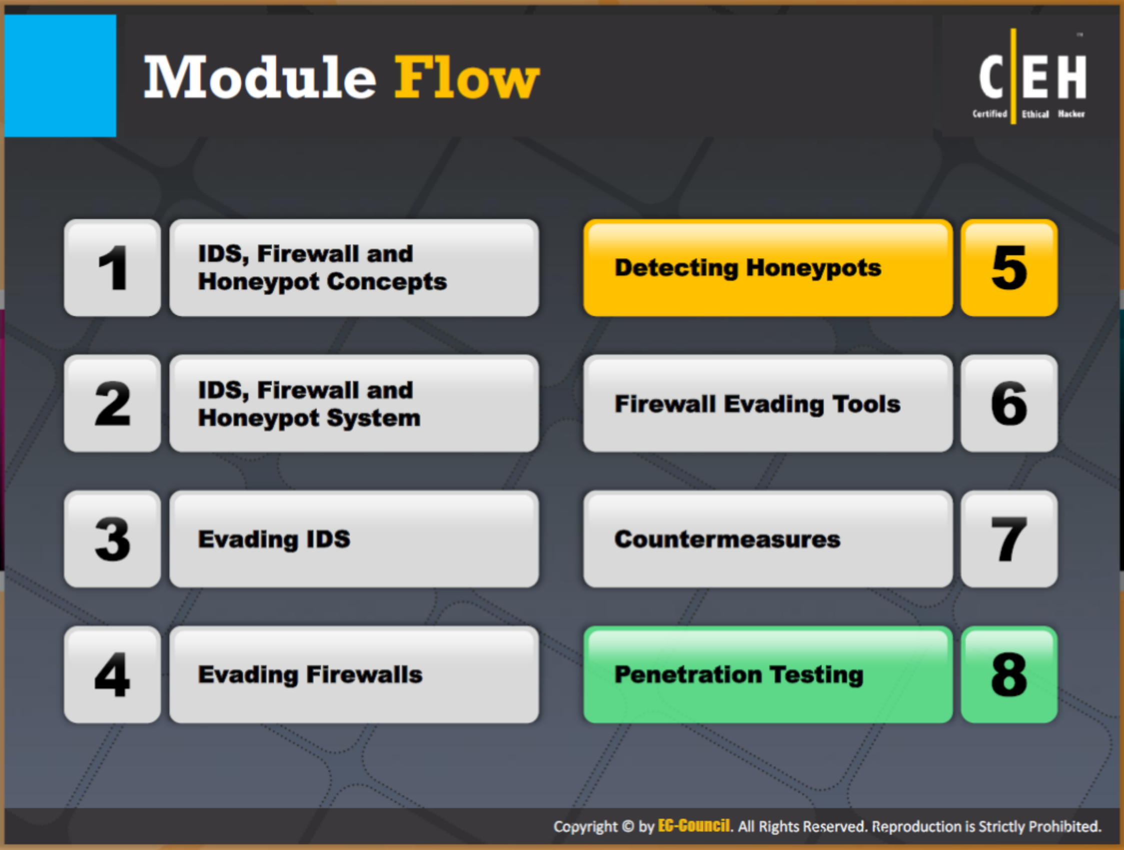

Corporate Network
DMZ
User C
User B
User A
1
2
3
4
5
6
Juggyboy Server
192.76.23.14
Malicious server
127.22.16.64
Attacker

FIGURE 17.35: Bypassing a Firewall through a MITM Attack



Module Flow

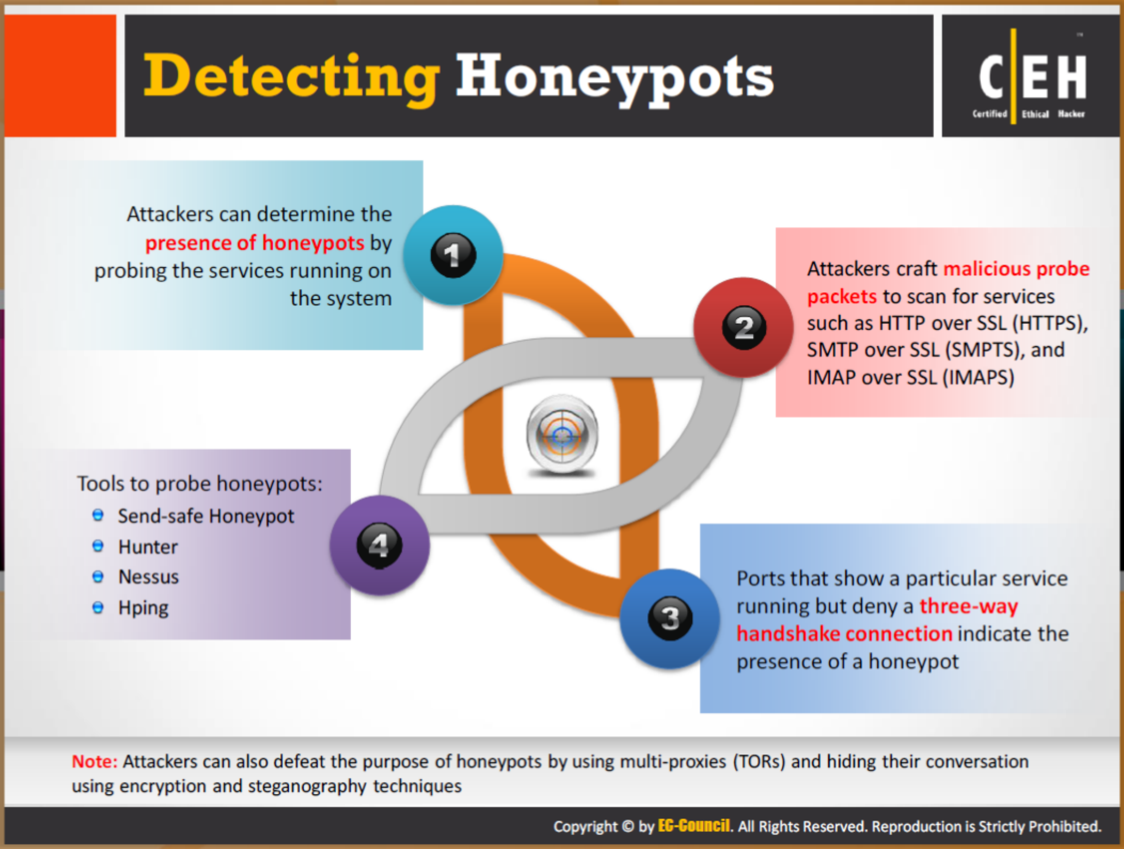
1. IDS, Firewall and Honeypot Concepts
2. IDS, Firewall and Honeypot System
3. Evading IDS
4. Evading Firewalls
5. Detecting Honeypots
6. Firewall Evading Tools
7. Countermeasures
8. Penetration Testing

Module Flow

Honeypots are the mechanisms intended to track or divert attackers from entering into a genuine network without adequate permissions. Attackers in an attempt to break into the target network first check for honeypots, if any are installed on the target network. Attackers perform honeypot detection to check whether the target network has a honeypot or not.

IDS, Firewall and Honeypot Concepts	Detecting Honeypots
IDS, Firewall and Honeypot System	Firewall Evading Tools
Evading IDS	Countermeasure
Evading Firewall	Penetration Testing

This section provides insight into honeypot detection and the tools that can be used for detecting honeypots.



Detecting Honeypots

1. Attackers can determine the presence of honeypots by probing the services running on the system
2. Attackers craft malicious probe packets to scan for services such as HTTP over SSL (HTTPS), SMTP over SSL (SMPTS), and IMAP over SSL (IMAPS)
3. Ports that show a particular service running but deny a three-way handshake connection indicate the presence of a honeypot
4. Tools to probe honeypots:
   - Send-safe Honeypot
   - Hunter
   - Nessus
   - Hping

Note: Attackers can also defeat the purpose of honeypots by using multi-proxies (TORs) and hiding their conversation using encryption and steganography techniques

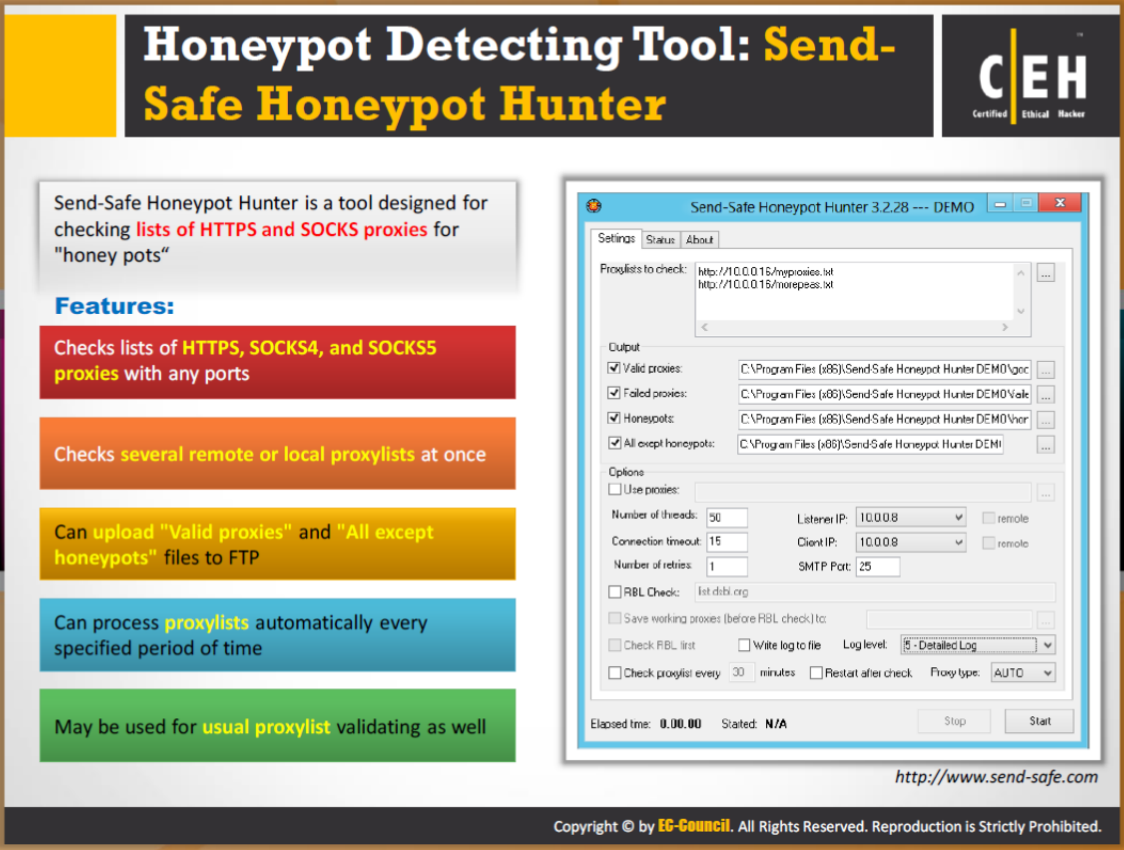
Detecting Honeypots

A honeypot is a system used on the Internet designed especially for diverting the attacker by tricking or attracting him or her when he or she attempts to gain unauthorized access to the information system in an organization. Just as honeypots are intended to divert the attackers from actual network, attackers use honeypot detection systems or methods to identify the honeypots installed on the target network. Once they detect honeypots, attackers try to bypass them so that they can focus on targeting the actual network. Detecting honeypots involves three basic steps:

- Attackers can determine the presence of honeypots by probing the services running on the system.
- Attackers craft malicious probe packets to scan for services such as HTTP over SSL (HTTPS), SMTP over SSL (SMPTS), and IMAP over SSL (IMAPS).
- Ports that show a particular service running but deny a three-way handshake connection indicate the presence of a honeypot.

Different tools such as Send-safe Honeypot, Hunter, Nessus, and Hping can be used for probing honeypots.

Note: Attackers can also defeat the purpose of honeypots by using multi-proxies (TORs) and hiding their conversation using encryption and steganography techniques.



Honeypot Detecting Tool: Send-Safe Honeypot Hunter

Send-Safe Honeypot Hunter is a tool designed for checking lists of HTTPS and SOCKS proxies for "honey pots"

Features:

- Checks lists of HTTPS, SOCKS4, and SOCKS5 proxies with any ports
- Checks several remote or local proxylists at once
- Can upload "Valid proxies" and "All except honeypots" files to FTP
- Can process proxylists automatically every specified period of time
- May be used for usual proxylist validating as well

Send-Safe Honeypot Hunter 3.2.28 --- DEMO

Settings | Status | About

Proxylists to check:
http://10.0.0.16/myproxies.txt
http://10.0.0.16/morepeas.txt

Output
☑ Valid proxies: C:\Program Files (x86)\Send-Safe Honeypot Hunter DEMO\goc
☑ Failed proxies: C:\Program Files (x86)\Send-Safe Honeypot Hunter DEMO\faile
☑ Honeypots: C:\Program Files (x86)\Send-Safe Honeypot Hunter DEMO\hor
☑ All exept honeypots: C:\Program Files (x86)\Send-Safe Honeypot Hunter DEM

Options
☐ Use proxies:
Number of threads: 50
Connection timeout: 15
Number of retries: 1
Listener IP: 10.0.0.8 ☐ remote
Client IP: 10.0.0.8 ☐ remote
SMTP Port: 25
☐ RBL Check: list.dsbl.org
☐ Save working proxies (before RBL check) to:
☐ Check RBL first
☐ Write log to file
Log level: 5 - Detailed Log
☐ Check proxylist every 30 minutes
☐ Restart after check
Proxy type: AUTO

Elapsed time: 0.00.00 Started: N/A

http://www.send-safe.com

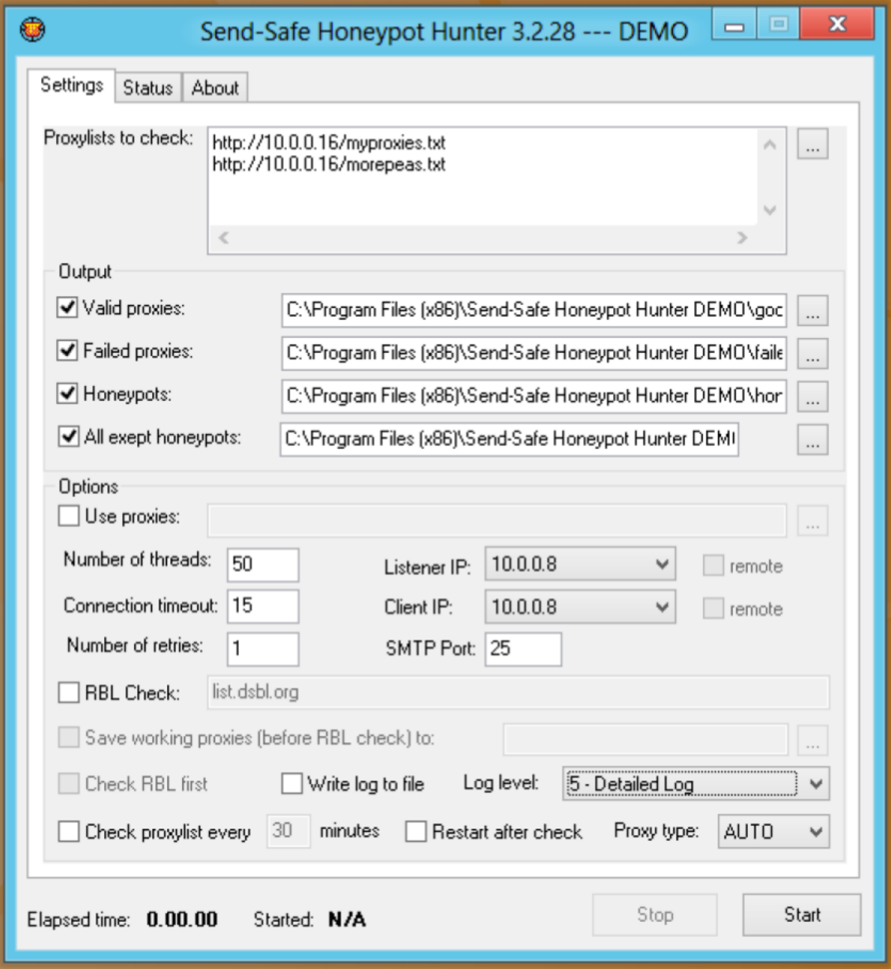
Honeypot Detecting Tool: Send-Safe Honeypot Hunter

Source: http://www.send-safe.com

Send-Safe Honeypot Hunter is a honeypot detection tool designed for checking lists of HTTPS and **SOCKS proxies** for honeypots.

Some of the Send-Safe Honeypot Hunter features include:

- Checks lists of HTTPS, SOCKS4, and SOCKS5 proxies with any ports
- Can check several remote or local proxylists at once
- Can upload "Valid proxies" and "All exept honeypots" files to FTP
- Can process proxylists automatically every specified period of time
- May be used for usual proxylist validating as well



Proxylists to check: http://10.0.0.16/myproxies.txt
http://10.0.0.16/morepeas.txt

Output

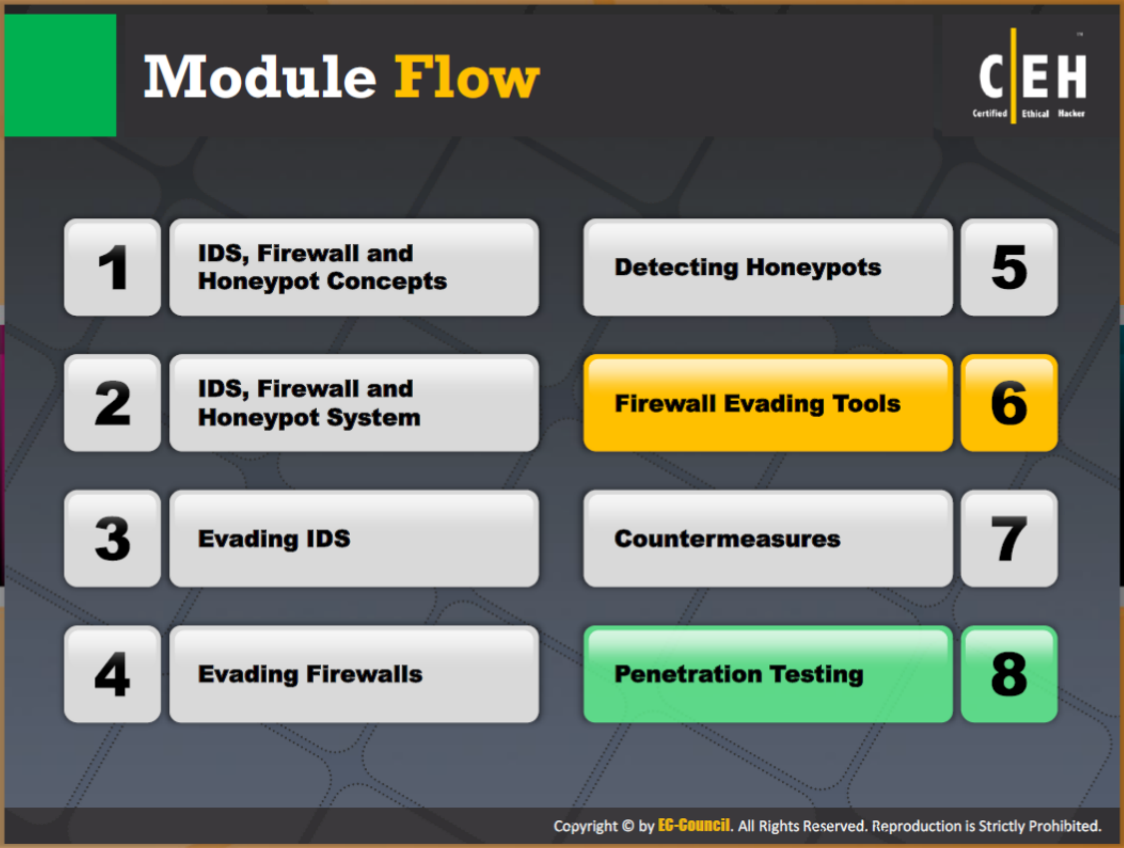
☑ Valid proxies: C:\Program Files (x86)\Send-Safe Honeypot Hunter DEMO\goc
☑ Failed proxies: C:\Program Files (x86)\Send-Safe Honeypot Hunter DEMO\faile
☑ Honeypots: C:\Program Files (x86)\Send-Safe Honeypot Hunter DEMO\hor
☑ All exept honeypots: C:\Program Files (x86)\Send-Safe Honeypot Hunter DEMI

Options

☐ Use proxies:
Number of threads: 50
Listener IP: 10.0.0.8 ☐ remote
Connection timeout: 15
Client IP: 10.0.0.8 ☐ remote
Number of retries: 1
SMTP Port: 25
☐ RBL Check: list.dsbl.org
☐ Save working proxies (before RBL check) to:
☐ Check RBL first
☐ Write log to file
Log level: 5 - Detailed Log
☐ Check proxylist every 30 minutes
☐ Restart after check
Proxy type: AUTO

Elapsed time: **0.00.00** Started: **N/A**

FIGURE 17.36: Honeypot Detecting Tool: Send-Safe Honeypot Hunter Screenshot



Module Flow

C|EH Certified Ethical Hacker

1. IDS, Firewall and Honeypot Concepts
2. IDS, Firewall and Honeypot System
3. Evading IDS
4. Evading Firewalls
5. Detecting Honeypots
6. Firewall Evading Tools
7. Countermeasures
8. Penetration Testing

Module Flow

Firewall evasion can be accomplished with the help of tools. These tools help an attacker in evading the firewall and thus breaking into the network. With the help of tools, an attacker can evade a firewall easily and also in less time.

IDS, Firewall and Honeypot Concepts	Detecting Honeypots
IDS, Firewall and Honeypot System	Firewall Evading Tools
Evading IDS	Countermeasure
Evading Firewall	Penetration Testing

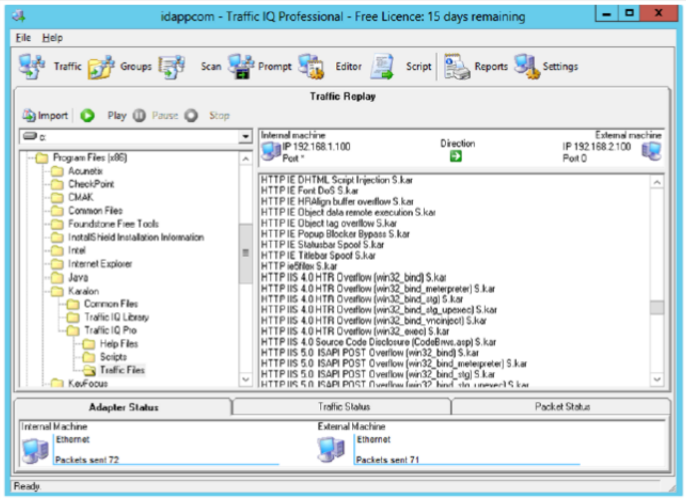
This section is dedicated to firewall evasion tools.

Firewall Evasion Tool: Traffic IQ Professional

- Traffic IQ Professional enables security professionals to audit and validate the behavior of security devices by generating the standard application traffic or attack traffic between two virtual machines

Traffic IQ Professional can be used to assess, audit, and test the behavioral characteristics of any non-proxy packet-filtering device including:

- Application layer firewalls
- Intrusion detection systems
- Intrusion prevention systems
- Routers and switches


idappcom - Traffic IQ Professional - Free Licence: 15 days remaining
File
Help
Traffic
Groups
Scan
Prompt
Editor
Script
Reports
Settings
Traffic Replay
Import
Play
Pause
Stop
c:
Program Files (x86)
Acunetix
CheckPoint
CMAK
Common Files
Foundstone Free Tools
InstallShield Installation Information
Intel
Internet Explorer
Java
Karalon
Common Files
Traffic IQ Library
Traffic IQ Pro
Help Files
Scripts
Traffic Files
KevFocus
Direction
Adapter Status
Traffic Status
Packet Status
Ready

Internal machine
IP 192.168.1.100
Port *

External machine
IP 192.168.2.100
Port 0

```
HTTP IE DHTML Script Injection S.kar
HTTP IE Font DoS S.kar
HTTP IE HRAlign buffer overflow S.kar
HTTP IE Object data remote execution S.kar
HTTP IE Object tag overflow S.kar
HTTP IE Popup Blocker Bypass S.kar
HTTP IE Statusbar Spoof S.kar
HTTP IE Titlebar Spoof S.kar
HTTP ie5filex S.kar
HTTP IIS 4.0 HTR Overflow (win32_bind) S.kar
HTTP IIS 4.0 HTR Overflow (win32_bind_meterpreter) S.kar
HTTP IIS 4.0 HTR Overflow (win32_bind_stg) S.kar
HTTP IIS 4.0 HTR Overflow (win32_bind_stg_upexec) S.kar
HTTP IIS 4.0 HTR Overflow (win32_bind_vncinject) S.kar
HTTP IIS 4.0 HTR Overflow (win32_exec) S.kar
HTTP IIS 4.0 Source Code Disclosure (CodeBrws.asp) S.kar
HTTP IIS 5.0 ISAPI POST Overflow (win32_bind) S.kar
HTTP IIS 5.0 ISAPI POST Overflow (win32_bind_meterpreter) S.kar
HTTP IIS 5.0 ISAPI POST Overflow (win32_bind_stg) S.kar
```

Internal Machine
Ethernet
Packets sent 72

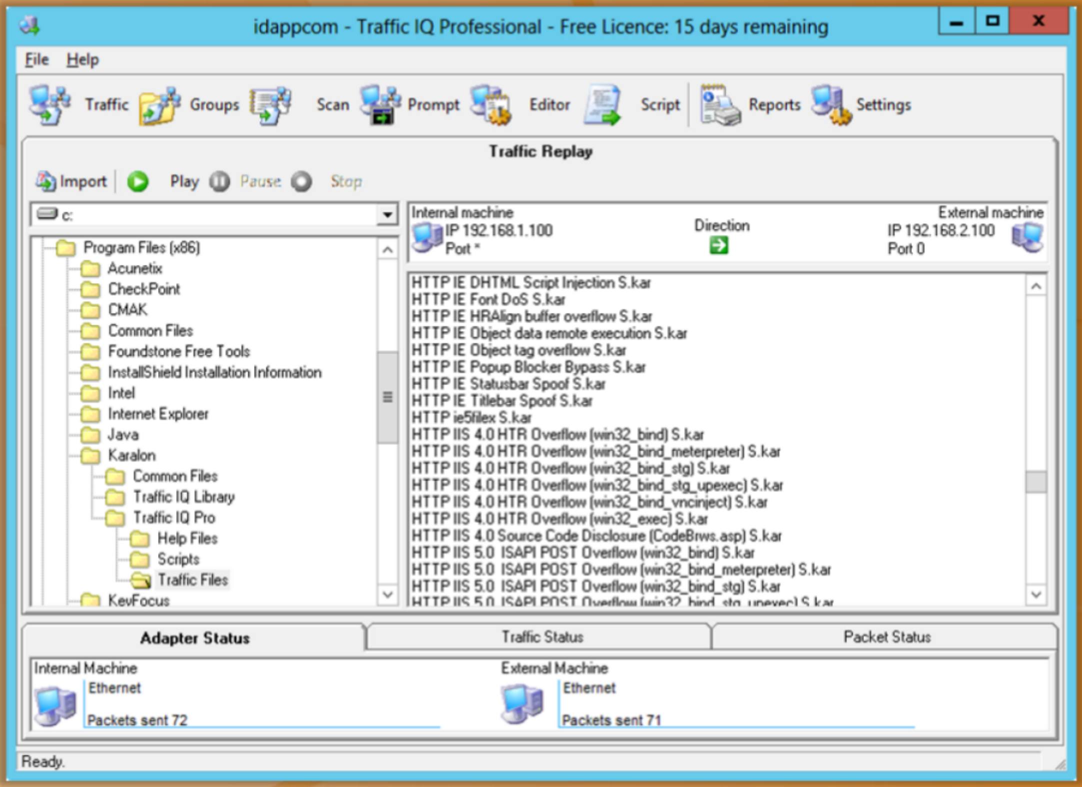
External Machine
Ethernet
Packets sent 71

http://www.idappcom.com

Firewall Evasion Tool: Traffic IQ Professional

Source: http://www.idappcom.com

Traffic IQ Professional **enables security professionals** to audit and validate the behavior of security devices by generating the standard application traffic or attack traffic between two virtual machines. The **unique features** and packet transmission capabilities of Traffic IQ Professional make the task of reliably auditing, validating, and proving **security compliance** easy and quick to complete. It can be used to assess, audit, and test the behavioral characteristics of any **non-proxy packet-filtering** device including **Application layer firewalls**, intrusion detection systems, intrusion prevention systems, and routers and switches.


idappcom - Traffic IQ Professional - Free Licence: 15 days remaining
File
Help
Traffic
Groups
Scan
Prompt
Editor
Script
Reports
Settings
Traffic Replay
Import
Play
Pause
Stop
c:
Program Files (x86)
Acunetix
CheckPoint
CMAK
Common Files
Foundstone Free Tools
InstallShield Installation Information
Intel
Internet Explorer
Java
Karalon
Common Files
Traffic IQ Library
Traffic IQ Pro
Help Files
Scripts
Traffic Files
KeyFocus
Direction
Adapter Status
Traffic Status
Packet Status

Internal machine
IP 192.168.1.100
Port *

External machine
IP 192.168.2.100
Port 0

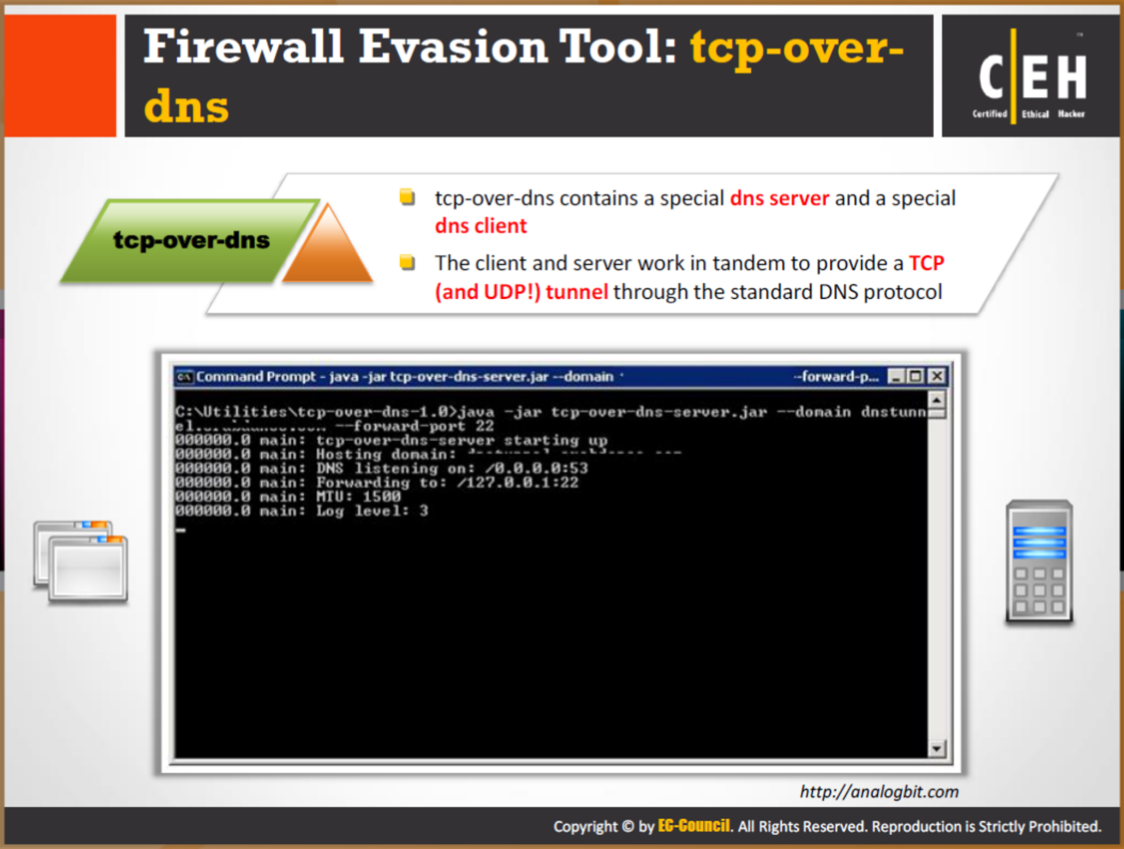
```
HTTP IE DHTML Script Injection S.kar
HTTP IE Font DoS S.kar
HTTP IE HRAlign buffer overflow S.kar
HTTP IE Object data remote execution S.kar
HTTP IE Object tag overflow S.kar
HTTP IE Popup Blocker Bypass S.kar
HTTP IE Statusbar Spoof S.kar
HTTP IE Titlebar Spoof S.kar
HTTP ie5filex S.kar
HTTP IIS 4.0 HTR Overflow (win32_bind) S.kar
HTTP IIS 4.0 HTR Overflow (win32_bind_meterpreter) S.kar
HTTP IIS 4.0 HTR Overflow (win32_bind_stg) S.kar
HTTP IIS 4.0 HTR Overflow (win32_bind_stg_upexec) S.kar
HTTP IIS 4.0 HTR Overflow (win32_bind_vncinject) S.kar
HTTP IIS 4.0 HTR Overflow (win32_exec) S.kar
HTTP IIS 4.0 Source Code Disclosure (CodeBrws.asp) S.kar
HTTP IIS 5.0  ISAPI POST Overflow (win32_bind) S.kar
HTTP IIS 5.0  ISAPI POST Overflow (win32_bind_meterpreter) S.kar
HTTP IIS 5.0  ISAPI POST Overflow (win32_bind_stg) S.kar
HTTP IIS 5.0  ISAPI POST Overflow (win32_bind_stg_upexec) S.kar
```

Internal Machine
Ethernet
Packets sent 72

External Machine
Ethernet
Packets sent 71

Ready.

FIGURE 17.37: Traffic IQ Professional Screenshot



Firewall Evasion Tool: tcp-over-dns

tcp-over-dns

- tcp-over-dns contains a special **dns server** and a special **dns client**
- The client and server work in tandem to provide a **TCP (and UDP!) tunnel** through the standard DNS protocol

Command Prompt - java -jar tcp-over-dns-server.jar --domain · --forward-p...

```
C:\Utilities\tcp-over-dns-1.0>java -jar tcp-over-dns-server.jar --domain dnstunn
el.[redacted] --forward-port 22
000000.0 main: tcp-over-dns-server starting up
000000.0 main: Hosting domain: [redacted]
000000.0 main: DNS listening on: /0.0.0.0:53
000000.0 main: Forwarding to: /127.0.0.1:22
000000.0 main: MTU: 1500
000000.0 main: Log level: 3
_
```

http://analogbit.com

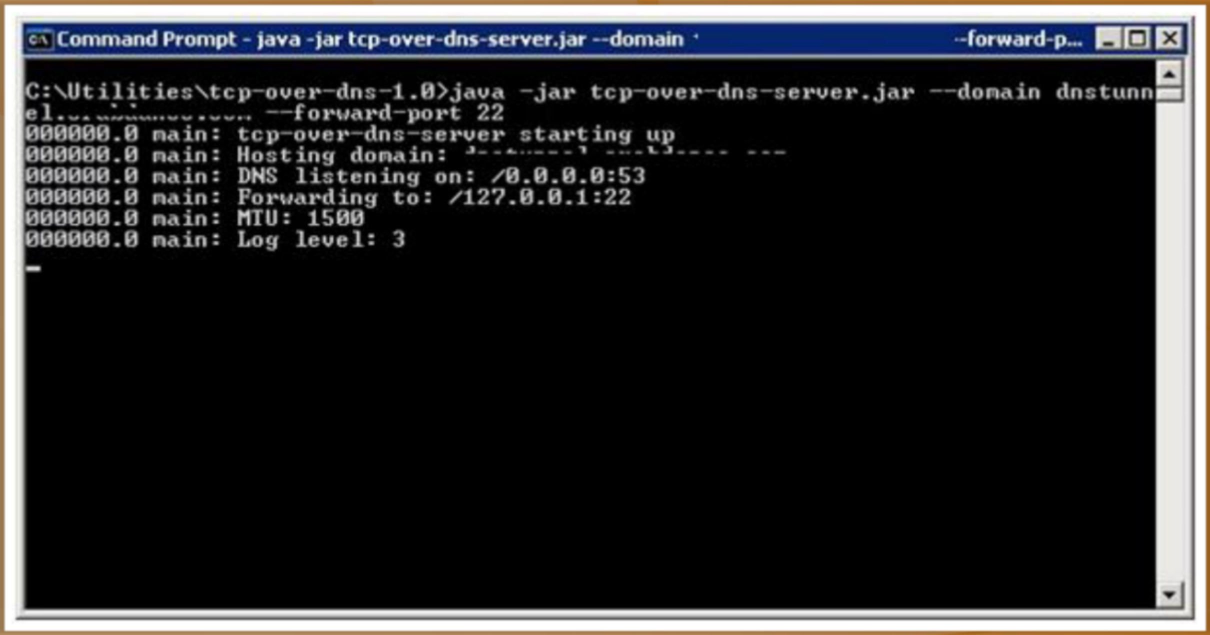
Firewall Evasion Tool: tcp-over-dns

Source: http://analogbit.com

tcp-over-dns contains a special dns server and a special **dns client**. The client and server work in tandem to provide a **TCP (and UDP!)** tunnel through the standard DNS protocol. It is similar to the defunct **NSTX dns tunneling software**. The purpose of this software is to succeed where **NSTX failed**. All NSTX tunnels disconnect within tens of seconds in real-world situations. tcp-over-dns is written to be quite robust while at the same time providing acceptable bandwidth speeds.

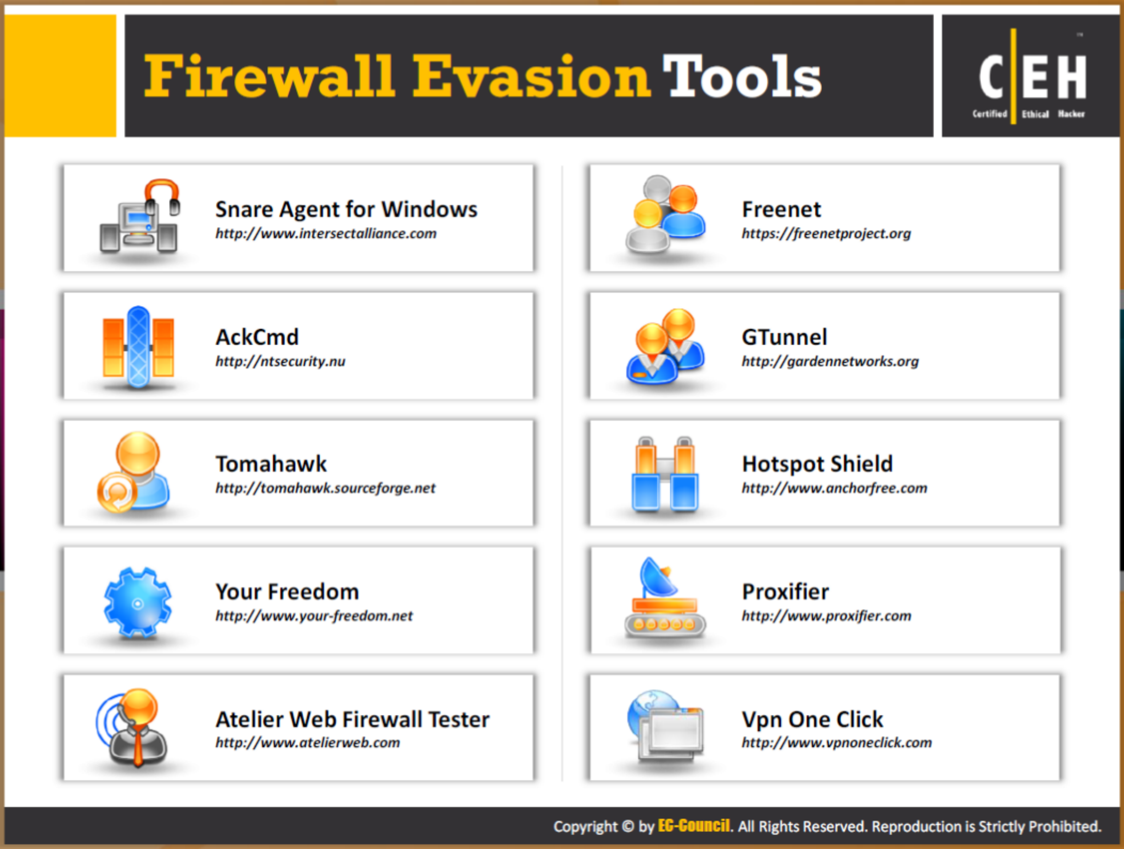
It features include:

- Windows, Linux, Solaris compatibility
- Sliding window packet transfers for increased speed and reliability
- Runtime selective LZMA compression
- TCP and UDP traffic tunneling


Command Prompt - java -jar tcp-over-dns-server.jar --domain [redacted] --forward-p...

```
C:\Utilities\tcp-over-dns-1.0>java -jar tcp-over-dns-server.jar --domain dnstunn
el.[redacted] --forward-port 22
000000.0 main: tcp-over-dns-server starting up
000000.0 main: Hosting domain: [redacted]
000000.0 main: DNS listening on: /0.0.0.0:53
000000.0 main: Forwarding to: /127.0.0.1:22
000000.0 main: MTU: 1500
000000.0 main: Log level: 3
_
```

FIGURE 17.38: tcp-over-dns in command promt



Firewall Evasion Tools

Snare Agent for Windows
http://www.intersectalliance.com

AckCmd
http://ntsecurity.nu

Tomahawk
http://tomahawk.sourceforge.net

Your Freedom
http://www.your-freedom.net

Atelier Web Firewall Tester
http://www.atelierweb.com

Freenet
https://freenetproject.org

GTunnel
http://gardennetworks.org

Hotspot Shield
http://www.anchorfree.com

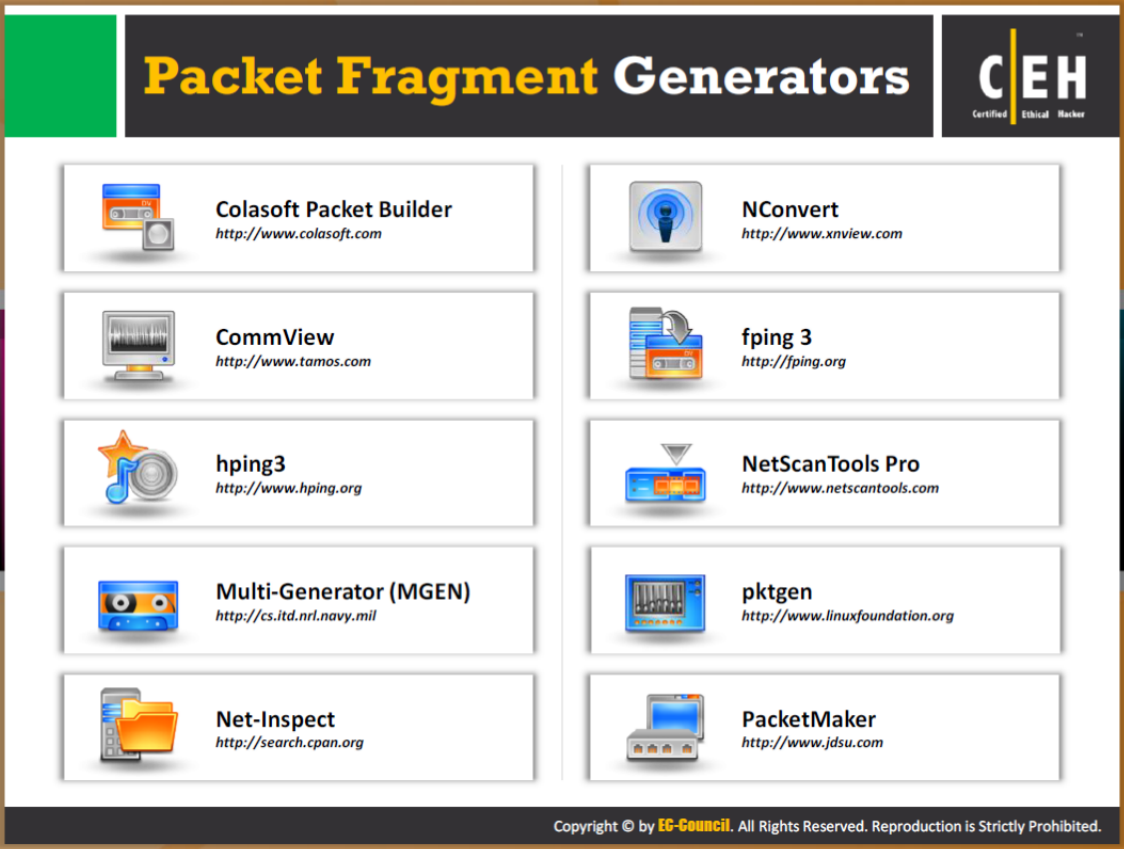
Proxifier
http://www.proxifier.com

Vpn One Click
http://www.vpnoneclick.com

Firewall Evasion Tools

Firewall evasion tools helps in **breaching a firewall** from inside as well as exporting data with innocent-looking packets that contain insufficient data for sniffers or firewalls to analyze. A few firewall evasion tools are listed as follows:

- Snare Agent for Windows available at http://www.intersectalliance.com
- AckCmd available at http://ntsecurity.nu
- Tomahawk available at http://tomahawk.sourceforge.net
- Your Freedom available at http://www.your-freedom.net
- Atelier Web Firewall Tester available at http://www.atelierweb.com
- Freenet available at https://freenetproject.org
- GTunnel available at http://gardennetworks.org
- Hotspot Shield available at http://www.anchorfree.com
- Proxifier available at http://www.proxifier.com
- Vpn One Click available at http://www.vpnoneclick.com



Packet Fragment Generators

Colasoft Packet Builder
http://www.colasoft.com

CommView
http://www.tamos.com

hping3
http://www.hping.org

Multi-Generator (MGEN)
http://cs.itd.nrl.navy.mil

Net-Inspect
http://search.cpan.org

NConvert
http://www.xnview.com

fping 3
http://fping.org

NetScanTools Pro
http://www.netscantools.com

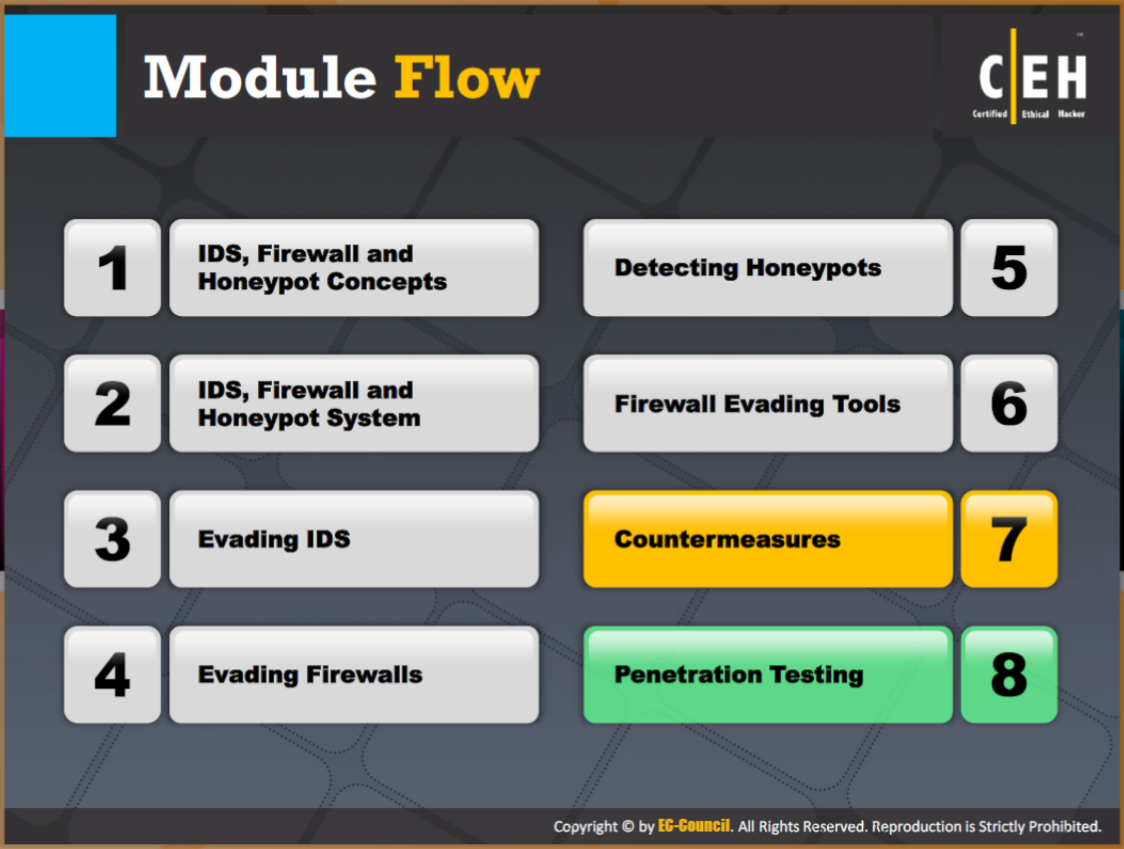
pktgen
http://www.linuxfoundation.org

PacketMaker
http://www.jdsu.com

Packet Fragment Generators

Packet fragment generators allow you to **edit** and **send packets** via your wireless network adapter. They allow you to hide your network file transfers across the Internet. By utilizing packet forgery, these tools hide your **file transfer** by cloaking it in seemingly harmless data. A few packet fragment generators are listed as follows:

- Colasoft Packet Builder available at http://www.colasoft.com
- CommView available at http://www.tamos.com
- hping3 available at http://www.hping.org
- Multi-Generator (MGEN) available at http://cs.itd.nrl.navy.mil
- Net-Inspect available at http://search.cpan.org
- NConvert available at http://www.xnview.com
- fping 3 available at http://fping.org
- NetScanTools Pro available at http://www.netscantools.com
- Pktgen available at http://www.linuxfoundation.org
- PacketMaker available at http://www.jdsu.com



Module Flow

1 IDS, Firewall and Honeypot Concepts

2 IDS, Firewall and Honeypot System

3 Evading IDS

4 Evading Firewalls

5 Detecting Honeypots

6 Firewall Evading Tools

7 Countermeasures

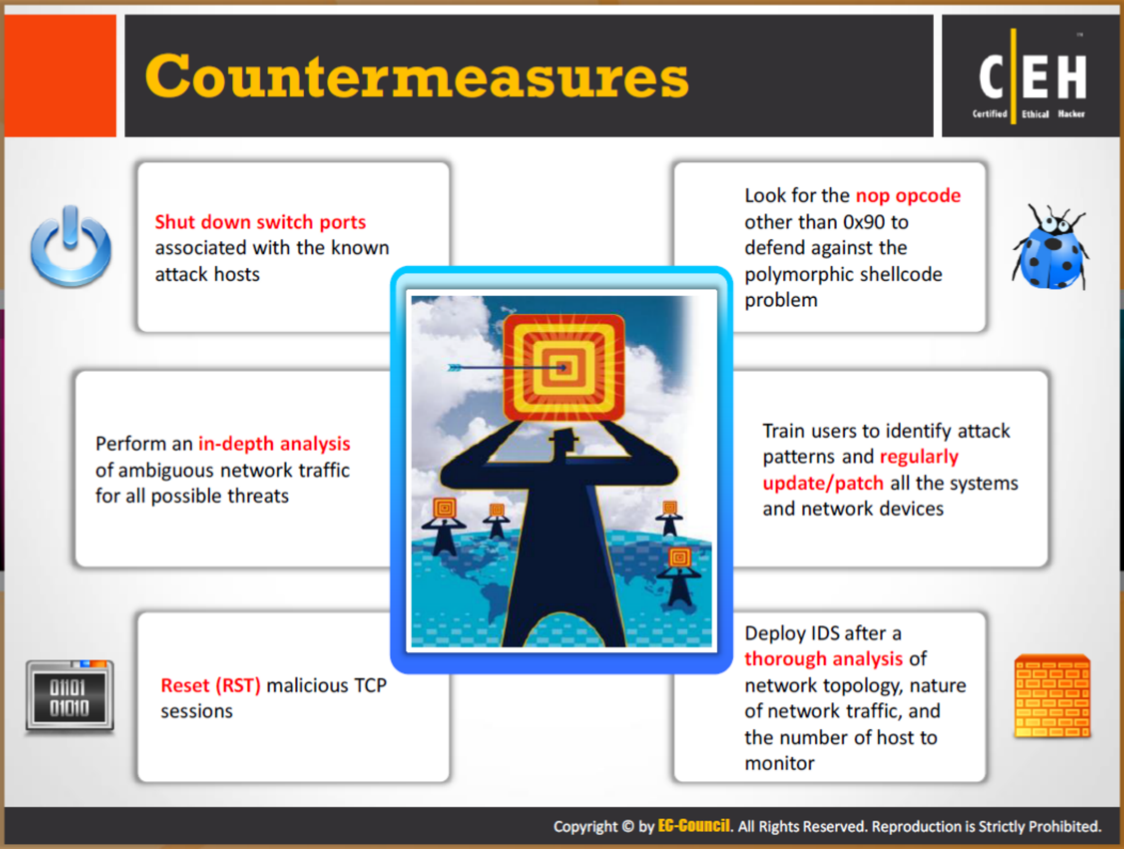
8 Penetration Testing

Module Flow

So far, we have discussed various concepts and topics related to intruding into or bypassing security mechanisms such as IDSes, firewalls, and honeypots. Now we will discuss the ways to protect them, i.e., countermeasures. Countermeasures help in enhancing security.

IDS, Firewall and Honeypot Concepts	Detecting Honeypots
IDS, Firewall and Honeypot System	Firewall Evading Tools
Evading IDS	Countermeasure
Evading Firewall	Penetration Testing

This section highlights various countermeasures against IDSes, firewalls, and honeypot attacks.



Countermeasures

Shut down switch ports associated with the known attack hosts

Look for the **nop opcode** other than 0x90 to defend against the polymorphic shellcode problem

Perform an **in-depth analysis** of ambiguous network traffic for all possible threats

Train users to identify attack patterns and **regularly update/patch** all the systems and network devices

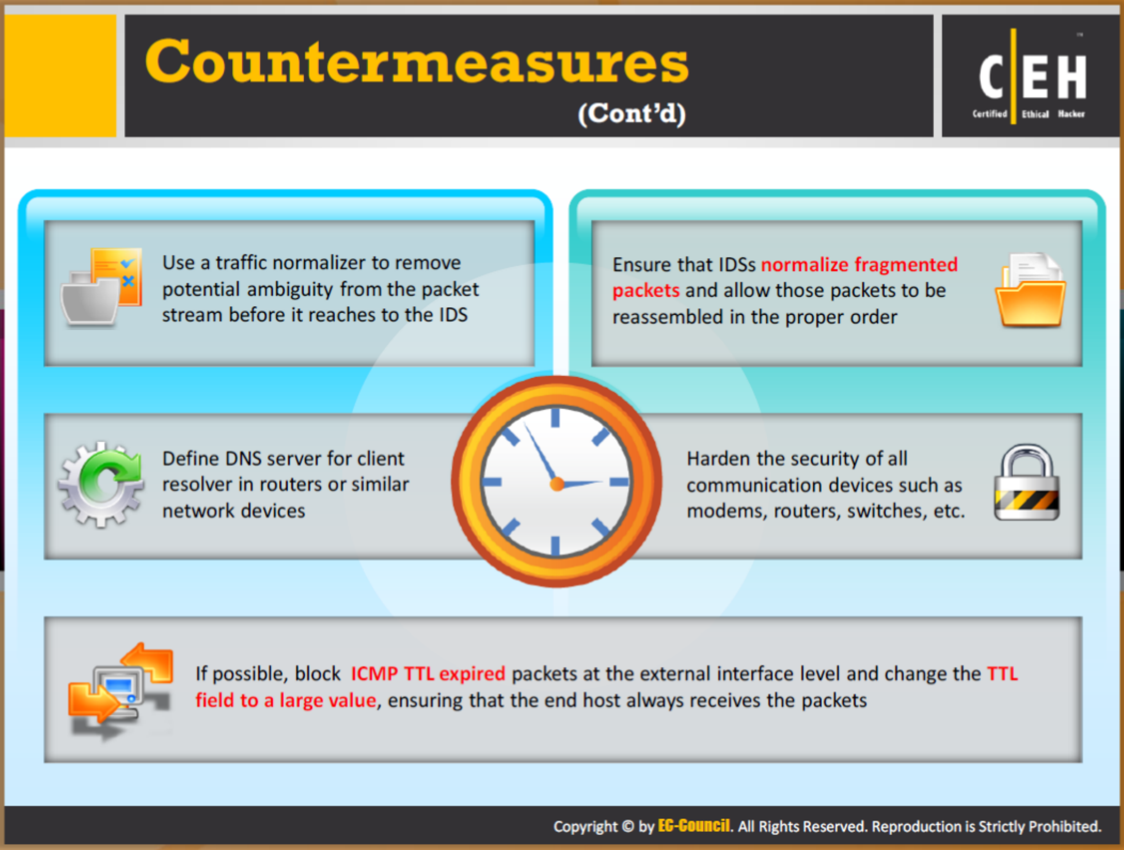
Reset (RST) malicious TCP sessions

Deploy IDS after a **thorough analysis** of network topology, nature of network traffic, and the number of host to monitor

Countermeasures

The following are few countermeasures that provide **protection against evading IDSes**, **firewalls**, and **honeypots**:

- Administratively shut down a switch port interface associated with a system from which attacks are being launched.
- Look for the nop opcode other than **0x90** to defend against the polymorphic **shellcode problem**.
- Perform "bifurcating analysis," in which the monitor deals with ambiguous traffic streams by instantiating **separate analysis** threads for each possible interpretation of the ambiguous traffic.
- Maintain security vulnerability awareness, patch vulnerabilities as soon as possible, and wisely choose the IDS based on the network topology and network traffic received.
- Generate TCP RST packets to tear down malicious TCP sessions, any issues of several available ICMP error code packets in response to malicious UDP traffic.
- Interact with the external firewall or router to add a general rule to block all communication from individual IP addresses or entire networks.



Countermeasures (Cont'd)

Use a traffic normalizer to remove potential ambiguity from the packet stream before it reaches to the IDS

Ensure that IDSs normalize fragmented packets and allow those packets to be reassembled in the proper order

Define DNS server for client resolver in routers or similar network devices

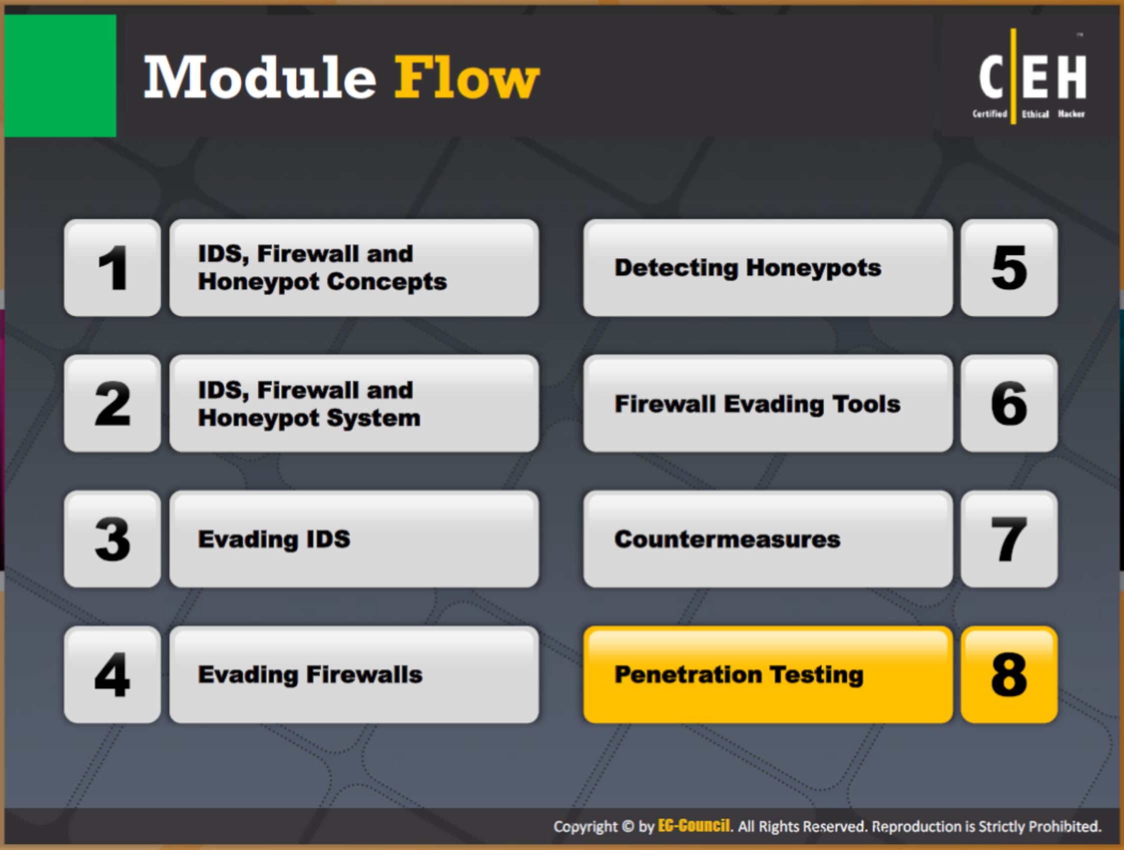
Harden the security of all communication devices such as modems, routers, switches, etc.

If possible, block ICMP TTL expired packets at the external interface level and change the TTL field to a large value, ensuring that the end host always receives the packets

Countermeasures (Cont'd)

The following are additional countermeasures against evading IDSes, firewalls, and honeypots:

- Implement a "**traffic normalizer**": a network forwarding element that attempts to eliminate ambiguous network traffic and reduce the amount of connection state that the monitor must maintain.
- Ensure that **IDSss normalize fragmented** packets and allow those packets to be reassembled in the proper order, which enables the IDS to look at the information just as the end host can see it.
- Keep updating the **IDS system** and **firewall software regularly**.
- Maintain security vulnerability awareness, patch vulnerabilities as soon as possible, and wisely choose the IDS based on the network topology and network traffic received.
- Change the TTL field to a large value, ensuring that the end host always receives the packets. In such case, attackers cannot slip information to the IDS. As a result, that data never reaches the end host, leaving the end host with the **malicious payload**.



Module Flow

1 IDS, Firewall and Honeypot Concepts

2 IDS, Firewall and Honeypot System

3 Evading IDS

4 Evading Firewalls

5 Detecting Honeypots

6 Firewall Evading Tools

7 Countermeasures

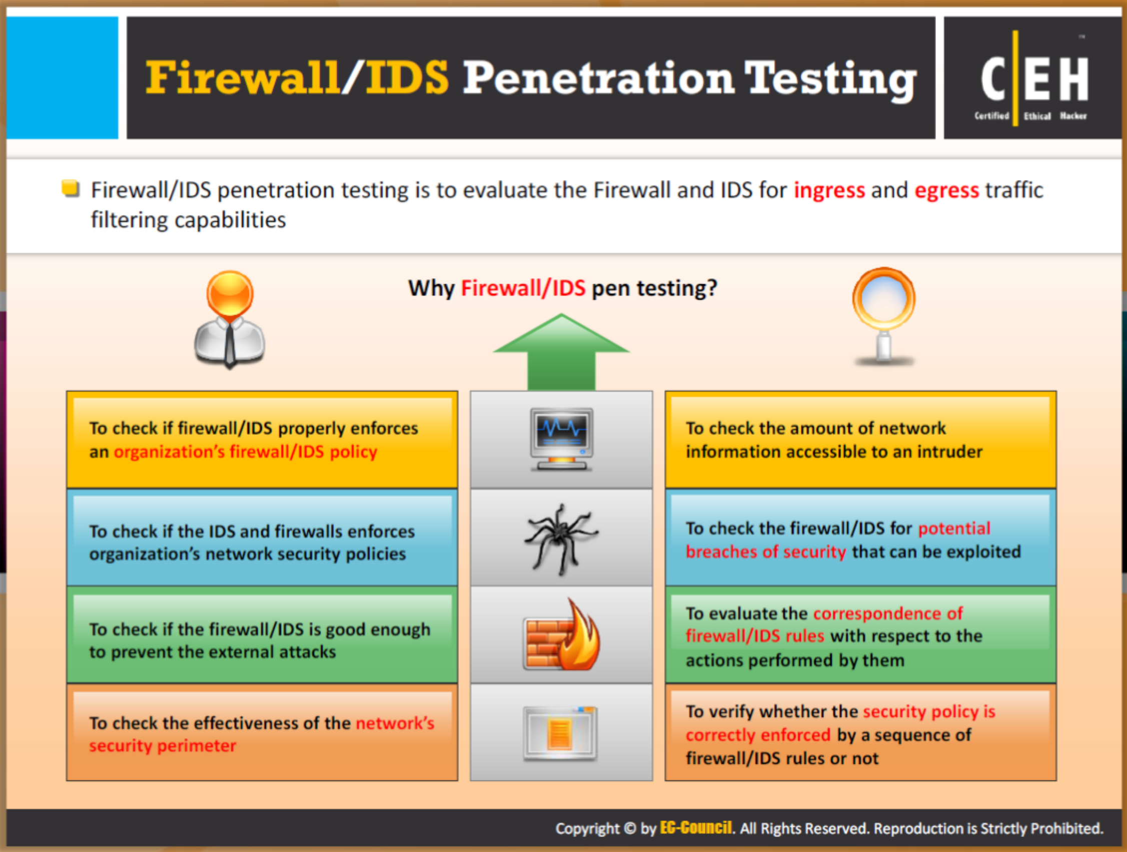
8 Penetration Testing

Module Flow

You need to conduct penetration test on firewalls, IDSes, and honeypots in order to ensure that they can withstand against different types attacks carried out by attackers. As a pen tester, you should conduct penetration testing on firewalls, IDSes, and honeypots to determine the vulnerabilities present in them before the attacker determines and exploits them.

IDS, Firewall and Honeypot Concepts	IDS, Firewall and Honeypot System
Evading IDS	Evading Firewalls
Detecting Honeypots	Firewall Evading Tools
Coutermeasures	Penetration Testing

This section shows the importance of firewall/IDS pen testing and also describes the steps involved in it.



Firewall/IDS Penetration Testing

- Firewall/IDS penetration testing is to evaluate the Firewall and IDS for ingress and egress traffic filtering capabilities

Why Firewall/IDS pen testing?

- To check if firewall/IDS properly enforces an organization's firewall/IDS policy
- To check if the IDS and firewalls enforces organization's network security policies
- To check if the firewall/IDS is good enough to prevent the external attacks
- To check the effectiveness of the network's security perimeter
- To check the amount of network information accessible to an intruder
- To check the firewall/IDS for potential breaches of security that can be exploited
- To evaluate the correspondence of firewall/IDS rules with respect to the actions performed by them
- To verify whether the security policy is correctly enforced by a sequence of firewall/IDS rules or not

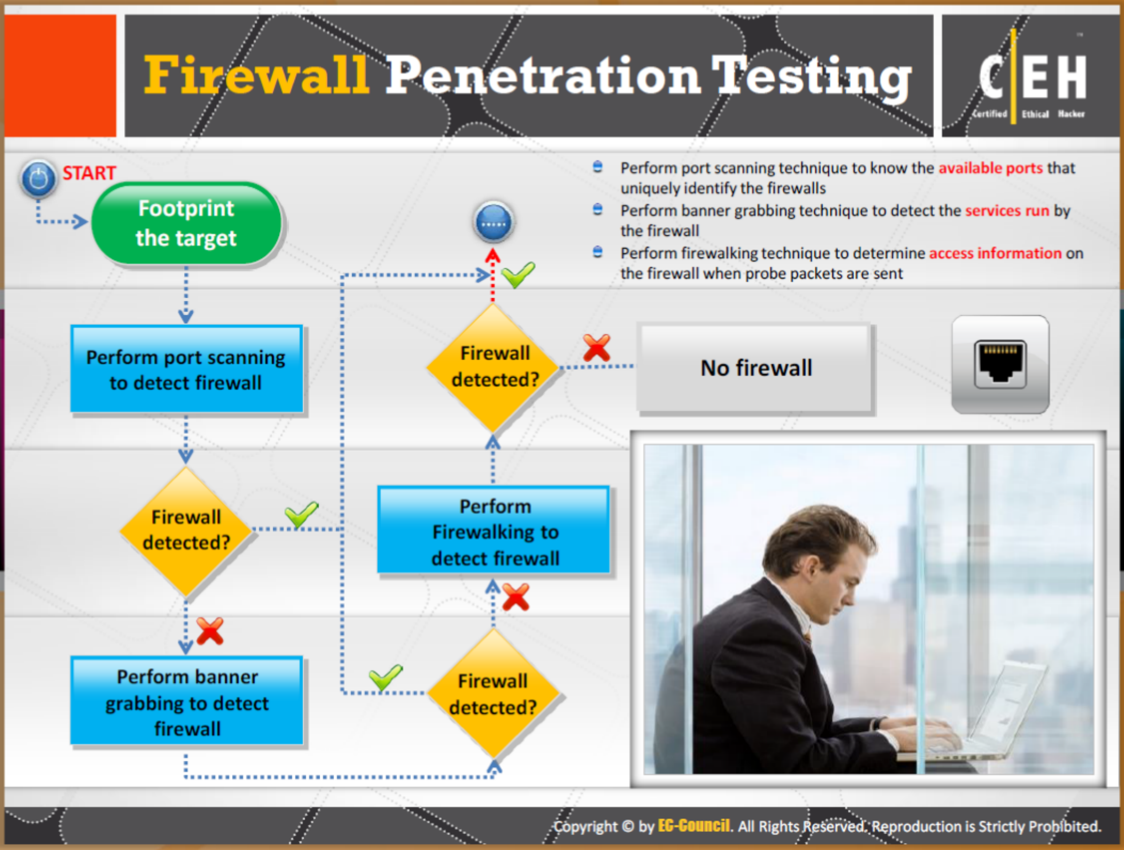
Firewall/IDS Penetration Testing

Firewall/IDS penetration testing is conducted to identify if there is any **security vulnerability** related to hardware, software and its configuration, and how to protect the network from outside attackers. It helps in **evaluating security** by testing for ingress and egress **vulnerabilities** and proper rule sets of the entire network with respect to the possibility of entry from an external location

Why firewall/IDS pen testing?

Firewall/IDS pen testing is required to:

- Check if firewall/IDS properly enforces an organization's firewall/IDS policy
- Check if firewall/IDS and components within network properly enforce an organization's network security policy
- Check the strength of **firewall/IDS protection** against externally initiated attacks
- Check the effectiveness of the network's security perimeter
- Check how much information about a network is available from outside a network
- Check the firewall/IDS for potential breaches of security that can be exploited
- Evaluate the correspondence of firewall/IDS rules with respect to the actions performed by them
- Verify whether the security policy is correctly enforced by a sequence of firewall/IDS rules or not



Firewall Penetration Testing

- Perform port scanning technique to know the available ports that uniquely identify the firewalls
- Perform banner grabbing technique to detect the services run by the firewall
- Perform firewalking technique to determine access information on the firewall when probe packets are sent

CEH
Certified Ethical Hacker
START
Footprint the target
Perform port scanning to detect firewall
Firewall detected?
Perform banner grabbing to detect firewall
Firewall detected?
Perform Firewalking to detect firewall
Firewall detected?
No firewall

Firewall Penetration Testing

As a pen tester, you should implement the following steps to conduct penetration testing on a firewall.

Step1: Footprint the target

You should footprint the target by **using various tools** such as Sam Spade, nslookup, traceroute, Nmap, and neotrace to learn about a system, its **remote access capabilities**, its ports and services, and the other aspects of its security.

Step2: Perform port scanning

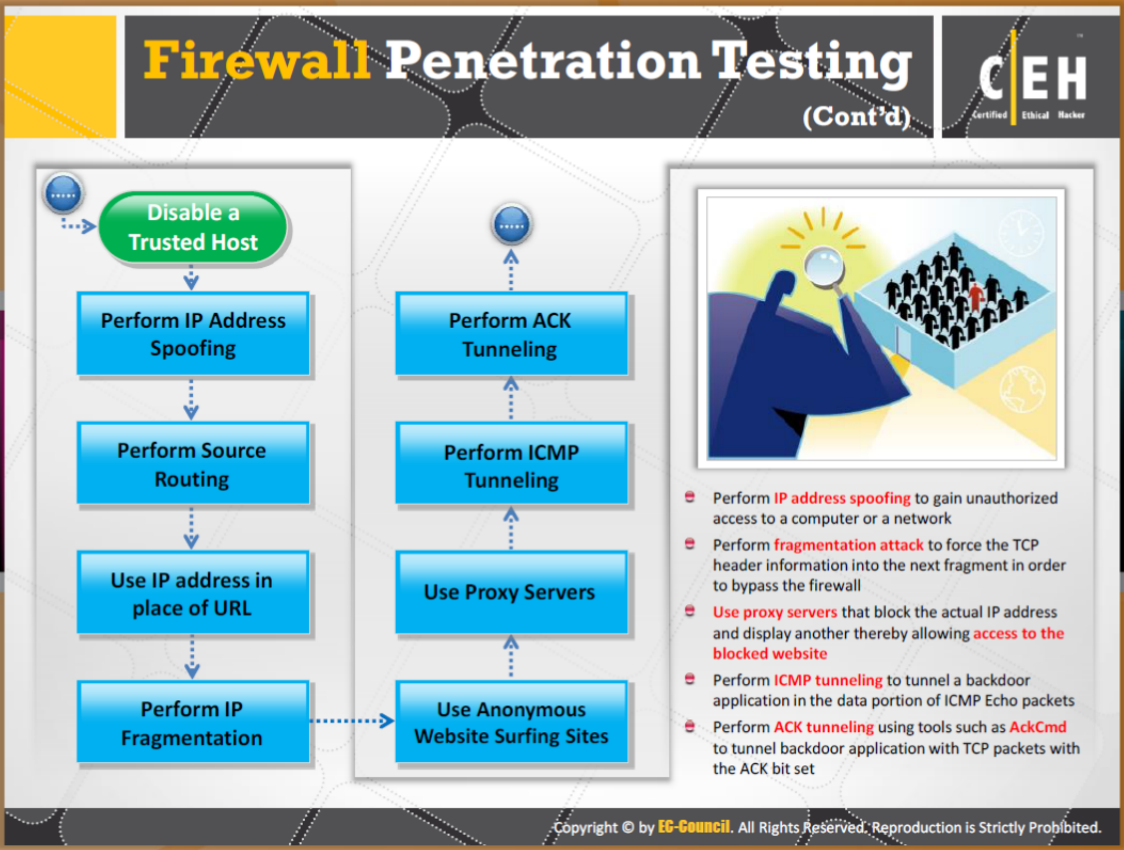
You should perform **port scanning** to detect the firewall to determine the available ports that uniquely identify the firewalls. If the firewall is detected, then disable a trusted host or perform banner grabbing to **detect the firewall**.

Step3: Perform banner grabbing

You should perform the **banner grabbing technique** to detect the services run by the firewall. If the firewall is detected, then disable a trusted host or perform **firewalking** to detect the firewall.

Step4: Perform firewalking

You should use the firewalking technique to determine access information on the firewall when probe packets are sent. If a firewall is detected, then disable a trusted host.



Firewall Penetration Testing (Cont'd)

- Disable a Trusted Host
- Perform IP Address Spoofing
- Perform Source Routing
- Use IP address in place of URL
- Perform IP Fragmentation
- Use Anonymous Website Surfing Sites
- Use Proxy Servers
- Perform ICMP Tunneling
- Perform ACK Tunneling

- Perform **IP address spoofing** to gain unauthorized access to a computer or a network
- Perform **fragmentation attack** to force the TCP header information into the next fragment in order to bypass the firewall
- **Use proxy servers** that block the actual IP address and display another thereby allowing **access to the blocked website**
- Perform **ICMP tunneling** to tunnel a backdoor application in the data portion of ICMP Echo packets
- Perform **ACK tunneling** using tools such as **AckCmd** to tunnel backdoor application with TCP packets with the ACK bit set

Firewall Penetration Testing (Cont'd)

Step 5: Disable the trusted host

Step6: Perform IP address spoofing

You should perform IP address spoofing to **gain unauthorized access** to a computer or a network.

Step 7: Perform source routing

Step8: Use an IP address in place of URL

Step 9: Perform a fragmentation attack

You should perform an IP fragmentation attack to force the **TCP header information** into the next fragment in order to bypass the firewall.

Step 10: Use anonymous website surfing sites

You should use anonymous website surfing sites to **hide your identity** from the Internet.

Step11: Use proxy servers

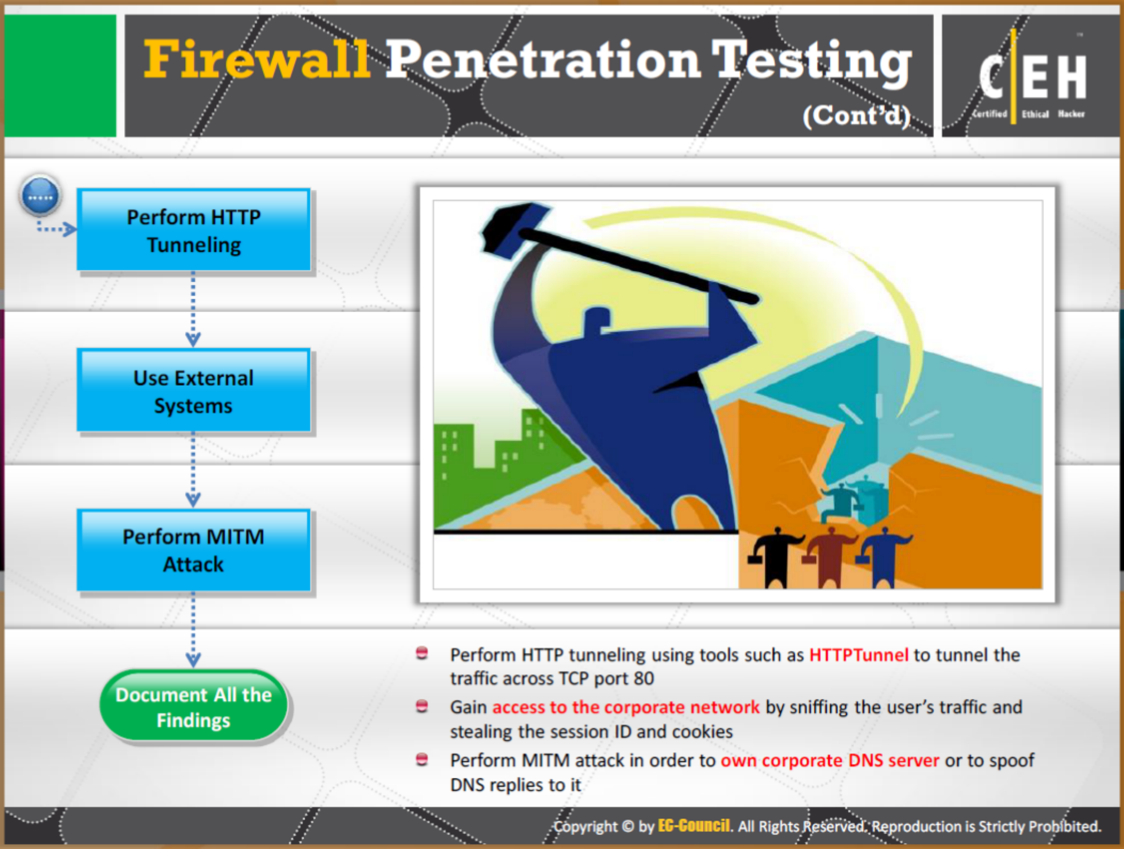
You should use proxy servers that block the actual IP address and display another, thereby allowing access to the blocked website.

Step12: Perform ICMP tunneling

You should perform ICMP tunneling to tunnel a backdoor application in the data portion of ICMP Echo packets.

Step13: Perform ACK tunneling

You should perform ACK tunneling using tools such as AckCmd to tunnel backdoor application with TCP packets with the ACK bit set.



Firewall Penetration Testing (Cont'd)

Perform HTTP Tunneling

Use External Systems

Perform MITM Attack

Document All the Findings

- Perform HTTP tunneling using tools such as HTTPTunnel to tunnel the traffic across TCP port 80
- Gain access to the corporate network by sniffing the user's traffic and stealing the session ID and cookies
- Perform MITM attack in order to own corporate DNS server or to spoof DNS replies to it

Firewall Penetration Testing (Cont'd)

Step14: Perform HTTP tunneling

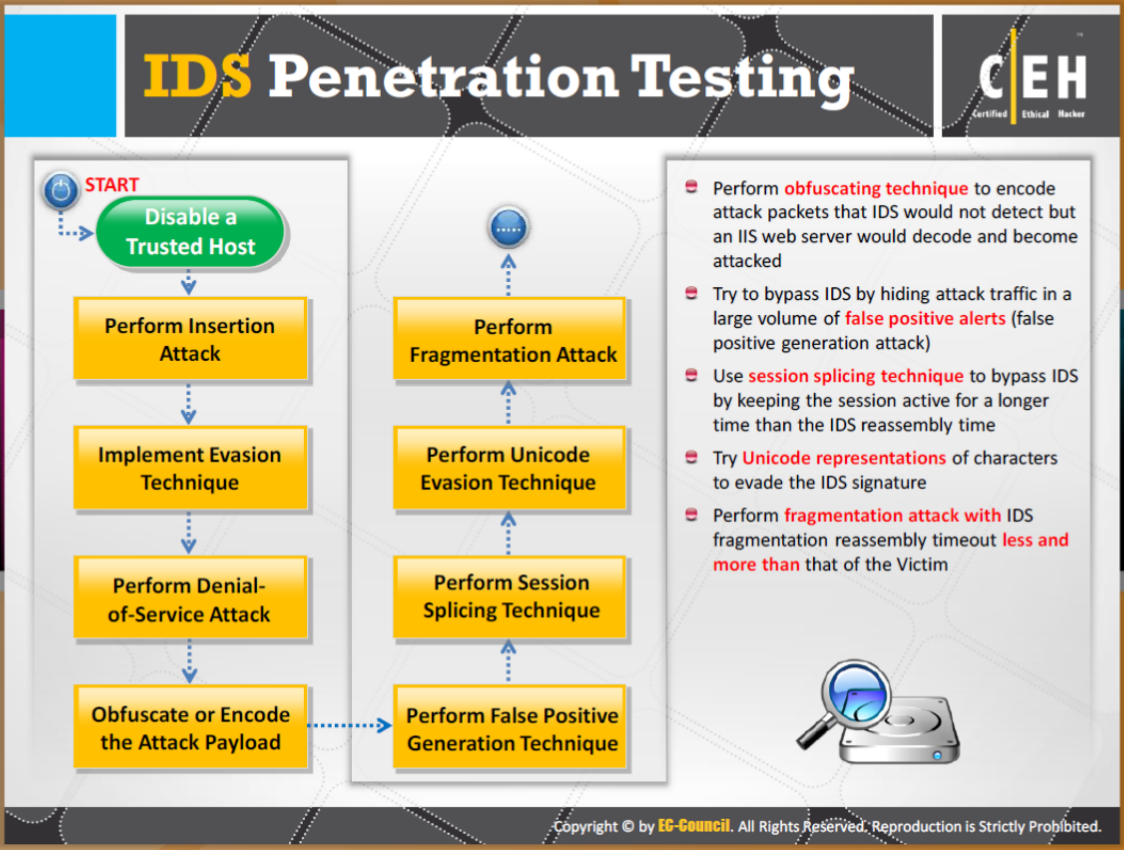
You should perform HTTP tunneling using tools such as **HTTPTunnel** to tunnel the traffic across TCP port 80.

Step15: Use external systems

Step16: Perform MITM Attack

You should perform an MITM attack in order to own corporate the **DNS server** or to **spoof DNS** replies to it.

Step 17: Document all the findings



IDS Penetration Testing

START

- Disable a Trusted Host
- Perform Insertion Attack
- Implement Evasion Technique
- Perform Denial-of-Service Attack
- Obfuscate or Encode the Attack Payload
- Perform False Positive Generation Technique
- Perform Session Splicing Technique
- Perform Unicode Evasion Technique
- Perform Fragmentation Attack

- Perform obfuscating technique to encode attack packets that IDS would not detect but an IIS web server would decode and become attacked
- Try to bypass IDS by hiding attack traffic in a large volume of false positive alerts (false positive generation attack)
- Use session splicing technique to bypass IDS by keeping the session active for a longer time than the IDS reassembly time
- Try Unicode representations of characters to evade the IDS signature
- Perform fragmentation attack with IDS fragmentation reassembly timeout less and more than that of the Victim

IDS Penetration Testing

You should carry out following steps to conduct IDS penetration testing.

Step1: Disable a trusted host

You should try to find and **disable the trusted host** so that the targeted host thinks that the traffic that the attacker will generate emanates from there.

Step2: Perform an insertion attack

Step3: Implement the evasion technique

Step4: Perform a denial-of-service attack

Step5: Obfuscate or encode the attack payload

You should implement the **obfuscating technique** to encode attack packets that the IDS would not detect but an IIS web server would decode and be attacked.

Step6: Perform the false positive generation technique

You should use the **false positive generation technique** to create a great deal of log "noise" in an attempt to blend real attacks with the false.

Step7: Perform the Session Splicing Technique

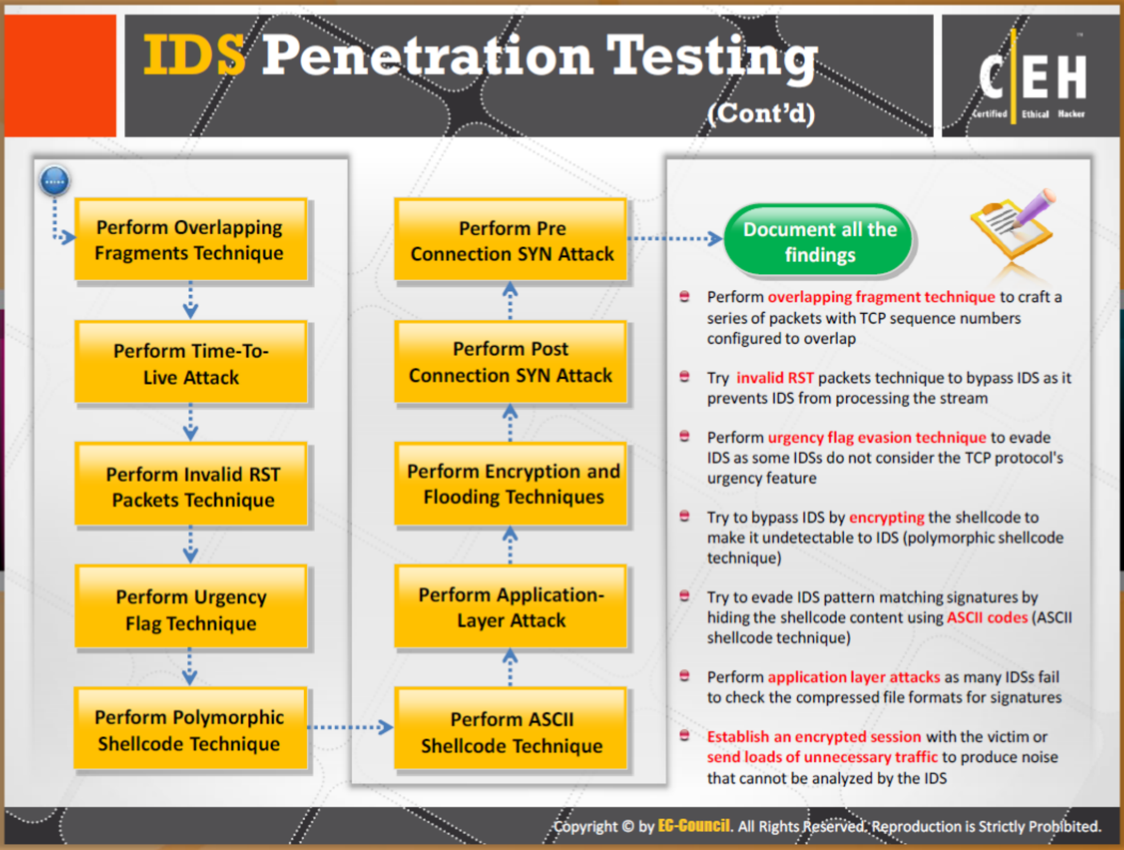
You should implement the **session splicing technique** to stop the IDS by keeping the session active longer than IDS will spend on reassembling it.

Step8: Perform the Unicode evasion technique

You should implement the **Unicode evasion technique** to evade IDSes as it is possible to have multiple representations of a single character.

Step 9: Perform a fragmentation attack

You should perform a fragmentation attack with **IDS fragmentation** reassembly timeout less and more than that of the victim.



IDS Penetration Testing (Cont'd)

- Perform Overlapping Fragments Technique
- Perform Time-To-Live Attack
- Perform Invalid RST Packets Technique
- Perform Urgency Flag Technique
- Perform Polymorphic Shellcode Technique
- Perform ASCII Shellcode Technique
- Perform Application-Layer Attack
- Perform Encryption and Flooding Techniques
- Perform Post Connection SYN Attack
- Perform Pre Connection SYN Attack
- Document all the findings

- Perform overlapping fragment technique to craft a series of packets with TCP sequence numbers configured to overlap
- Try invalid RST packets technique to bypass IDS as it prevents IDS from processing the stream
- Perform urgency flag evasion technique to evade IDS as some IDSs do not consider the TCP protocol's urgency feature
- Try to bypass IDS by encrypting the shellcode to make it undetectable to IDS (polymorphic shellcode technique)
- Try to evade IDS pattern matching signatures by hiding the shellcode content using ASCII codes (ASCII shellcode technique)
- Perform application layer attacks as many IDSs fail to check the compressed file formats for signatures
- Establish an encrypted session with the victim or send loads of unnecessary traffic to produce noise that cannot be analyzed by the IDS

IDS Penetration Testing (Cont'd)

Step10: Perform the overlapping fragments technique

You should use othe verlapping fragments technique to craft a series of packets with TCP sequence numbers configured to overlap.

Step 11: Perform a Time-To-Live attack

Step 12: Perform the invalid RST packets technique

You should use the invalid RST packets technique to evade detection by sending RST packets with an invalid checksum that causes the IDS to stop processing the stream.

Step13: Perform the urgency flag technique

You should use the urgency flag technique to evade IDSrd as some IDSrds do not consider the TCP protocol's urgency feature.

Step14: Perform the polymorphic shellcode technique

You should use the polymorphic shellcode technique to hide the shellcode by encrypting it in a simplistic form that is difficult for IDS to identify that data as a shellcode.

Step15: Perform the ASCII shellcode technique

You should perform the **ASCII shellcode** technique to bypass IDS pattern matching signatures because strings are hidden within the shellcode as in a polymorphic shellcode.

Step16: Perform an Application-layer attacks

You should try to perform Application-level attacks as many IDSes will have no way to check the compressed file format for signatures.

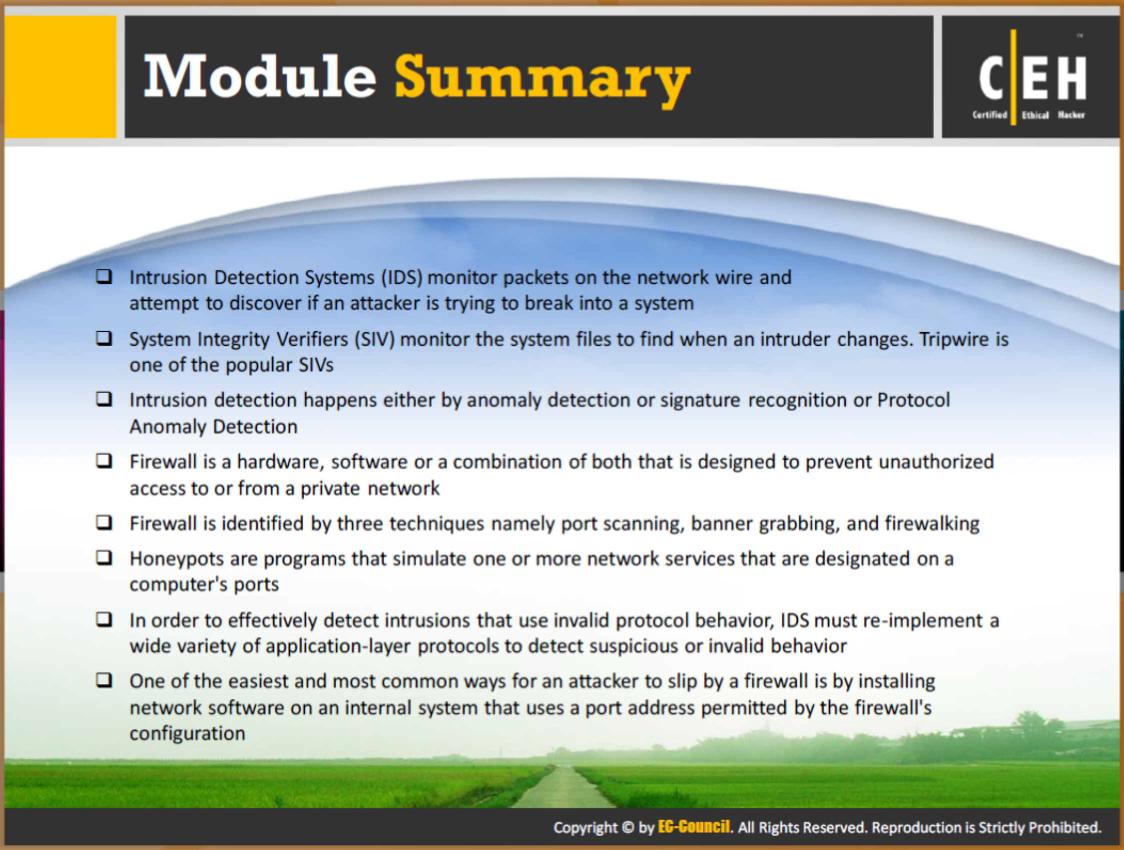
Step17: Perform encryption and flooding techniques

You should try **encryption** and **flooding attacks** with the victim or send loads of unnecessary traffic to produce noise that can't be analyzed by the IDS.

Step18: Perform a post-connection SYN attack

Step19: Perform a pre-connection SYN attack

Step 20: Document all the results obtained from this test



Module Summary

- Intrusion Detection Systems (IDS) monitor packets on the network wire and attempt to discover if an attacker is trying to break into a system
- System Integrity Verifiers (SIV) monitor the system files to find when an intruder changes. Tripwire is one of the popular SIVs
- Intrusion detection happens either by anomaly detection or signature recognition or Protocol Anomaly Detection
- Firewall is a hardware, software or a combination of both that is designed to prevent unauthorized access to or from a private network
- Firewall is identified by three techniques namely port scanning, banner grabbing, and firewalking
- Honeypots are programs that simulate one or more network services that are designated on a computer's ports
- In order to effectively detect intrusions that use invalid protocol behavior, IDS must re-implement a wide variety of application-layer protocols to detect suspicious or invalid behavior
- One of the easiest and most common ways for an attacker to slip by a firewall is by installing network software on an internal system that uses a port address permitted by the firewall's configuration

Module Summary

- Intrusion detection systems (IDSes) monitor packets on the network wire and attempt to discover if an attacker is trying to break into a system.
- System integrity verifiers (SIVs) monitor the system files to find when an intruder changes. Tripwire is one of the popular SIVs.
- Intrusion detection happens either by anomaly detection or signature recognition or protocol anomaly detection.
- A firewall is hardware, software, or a combination of both that is designed to prevent unauthorized access to or from a private network.
- A firewall is identified by three techniques: port scanning, banner grabbing, and firewalking.
- Honeypots are programs that simulate one or more network services that are designated on a computer's ports.
- In order to effectively detect intrusions that use invalid protocol behavior, an IDS must re-implement a wide variety of Application-layer protocols to detect suspicious or invalid behavior.

- One of the easiest and most common ways for an attacker to slip by a firewall is by installing network software on an internal system that uses a port address permitted by the firewall's configuration.